SOFTWARE TOOL ARTICLE

REVISED An international virtual hackathon to build tools for the analysis of structural variants within species ranging from coronaviruses to vertebrates [version 2; peer review: 1 approved, 3 approved with reservations]

Ann M. Mc Cartney[1], Medhat Mahmoud[2], Michael Jochum[2], Daniel Paiva Agustinho[3], Barry Zorman[2], Ahmad Al Khleifat[4], Fawaz Dabbaghie[5], Rupesh K Kesharwani[2], Moritz Smolka[6], Moez Dawood[2], Dreycey Albin[7], Elbay Aliyev[8], Hakeem Almabrazi[8], Ahmed Arslan[9], Advait Balaji[10], Sairam Behera[2], Kimberley Billingsley[1], Daniel L Cameron[11], Joyjit Daw[12], Eric T. Dawson[12], Wouter De Coster[13], Haowei Du[2], Christopher Dunn[14], Rocio Esteban[15], Angad Jolly[2], Divya Kalra[2], Chunxiao Liao[10], Yunxi Liu[10], Tsung-Yu Lu[16], James M Havrilla[17], Michael M Khayat[2], Maximillian Marin[18], Jean Monlong[19], Stephen Price[20], Alejandro Rafael Gener[2], Jingwen Ren[16], Sagayamary Sagayaradj[21], Nicolae Sapoval[10], Claude Sinner[22], Daniela C. Soto[21], Arda Soylev[23], Arun Subramaniyan[24], Najeeb Syed[8], Neha Tadimeti[12], Pamella Tater[25], Pankaj Vats[12], Justin Vaughn[26], Kimberly Walker[2], Gaojianyong Wang[27], Qiandong Zeng[28], Shangzhe Zhang[29], Tingting Zhao[30], Bryce Kille[10], Evan Biederstedt[18], Mark Chaisson[16], Adam English[2], Zev Kronenberg[14], Todd J. Treangen[10], Timothy Hefferon[1], Chen-Shan Chin[25], Ben Busby[25], Fritz J Sedlazeck[2]

[1]NIH, Washington, USA
[2]Baylor College of Medicine, Houston, USA
[3]Washington University in St. Louis School of Medicine, St. Louis, USA
[4]King's College London, London, UK
[5]Institute for Medical Biometry and Bioinformatics, Düsseldorf, Germany
[6]University of Vienna, Vienna, Austria
[7]University of Colorado at Boulder, Boulder, USA
[8]Sidra Medicine, Doha, Qatar
[9]Stanford University School of Medicine, California, USA
[10]Rice University, Houston, USA
[11]Walter and Eliza Hall Institute of Medical Research, Melbourne, Australia
[12]NVIDIA Corporation, Santa Clara, California, USA
[13]VIB Center for Molecular Neurology, Antwerp, Belgium
[14]Pacific Biosciences, Menlo Park, USA
[15]Oxford Nanopore Technologies Ltd, Oxford, UK

[16]USC, Los Angeles, USA
[17]Children's Hospital of Philadelphia, Philadelphia, USA
[18]Harvard Medical School, Boston, USA
[19]UC Santa Cruz Genomics Institute, Santa Cruz, USA
[20]Carnegie Mellon University, Pittsburgh, USA
[21]University of California, Davis, USA
[22]University of Texas at Dallas, Richardson, USA
[23]Konya Food and Agriculture University, Konya, Turkey
[24]University of Michigan, Ann Arbor, USA
[25]DNAnexus, Mountain View, USA
[26]USDA-ARS, Athens, USA
[27]Max Planck Institute for Molecular Genetics, Berlin, Germany
[28]Laboratory Corporation of America Holdings, Westborough, USA
[29]Lanzhou University, Lanzhou, China
[30]University of Pittsburgh, Pittsburgh, USA

v2 **First published:** 26 Mar 2021, **10**:246
https://doi.org/10.12688/f1000research.51477.1

Latest published: 03 Sep 2021, **10**:246
https://doi.org/10.12688/f1000research.51477.2

Abstract

In October 2020, 62 scientists from nine nations worked together remotely in the Second Baylor College of Medicine & DNAnexus hackathon, focusing on different related topics on Structural Variation, Pan-genomes, and SARS-CoV-2 related research. The overarching focus was to assess the current status of the field and identify the remaining challenges. Furthermore, how to combine the strengths of the different interests to drive research and method development forward. Over the four days, eight groups each designed and developed new open-source methods to improve the identification and analysis of variations among species, including humans and SARS-CoV-2. These included improvements in SV calling, genotyping, annotations and filtering. Together with advancements in benchmarking existing methods. Furthermore, groups focused on the diversity of SARS-CoV-2. Daily discussion summary and methods are available publicly at https://github.com/collaborativebioinformatics provides valuable insights for both participants and the research community.

Keywords

Structural variant, CNV, SARS-CoV-2, NextGeneration Sequencing

This article is included in the Sidra Medicine gateway.

Open Peer Review

Reviewer Status ✓ ? ? ?

	Invited Reviewers			
	1	2	3	4
version 2 (revision) 03 Sep 2021	✓ report			
version 1 26 Mar 2021	? report	? report	? report	? report

1. **Francois Sabot**, Univ Montpellier, IRD, CIRAD, Montpellier, France

2. **Kamil S. Jaron**, The University of Edinburgh, Edinburgh, UK

3. **Ricardo Assunção Vialle**, Icahn School of Medicine at Mount Sinai, New York, USA

4. **Loren L. Flynn**, Murdoch University, Murdoch, Australia

Any reports and responses or comments on the article can be found at the end of the article.

This article is included in the Disease Outbreaks gateway.

This article is included in the Coronavirus collection.

This article is included in the Max Planck Society collection.

This article is included in the Hackathons collection.

Corresponding authors: Ann M. Mc Cartney (ann.mccartney@nih.gov), Timothy Hefferon (timothy.hefferon@nih.gov), Ben Busby (bbusby@dnanexus.com), Fritz J Sedlazeck (fritz.sedlazeck@bcm.edu)

Author roles: Mc Cartney AM: Data Curation, Methodology, Software, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing; **Mahmoud M**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Jochum M**: Data Curation, Methodology, Project Administration, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Agustinho DP**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Zorman B**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Al Khleifat A**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Dabbaghie F**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **K Kesharwani R**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Smolka M**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Dawood M**: Data Curation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Albin D**: Data Curation, Methodology, Software; **Aliyev E**: Data Curation, Methodology, Software; **Almabrazi H**: Data Curation, Methodology, Software; **Arslan A**: Data Curation, Methodology, Software; **Balaji A**: Data Curation, Methodology, Software; **Behera S**: Data Curation, Methodology, Software; **Billingsley K**: Data Curation, Methodology, Software; **L Cameron D**: Data Curation, Methodology, Software; **Daw J**: Data Curation, Methodology, Software; **T. Dawson E**: Data Curation, Methodology, Software; **De Coster W**: Data Curation, Methodology, Software; **Du H**: Data Curation, Methodology, Software; **Dunn C**: Data Curation, Methodology, Software; **Esteban R**: Data Curation, Methodology, Software; **Jolly A**: Data Curation, Methodology, Software; **Kalra D**: Data Curation, Methodology, Software; **Liao C**: Data Curation, Methodology, Software; **Liu Y**: Data Curation, Methodology, Software; **Lu TY**: Data Curation, Methodology, Software; **M Havrilla J**: Data Curation, Methodology, Software; **M Khayat M**: Data Curation, Methodology, Software; **Marin M**: Data Curation, Methodology, Software; **Monlong J**: Data Curation, Methodology, Software; **Price S**: Data Curation, Methodology, Software; **Rafael Gener A**: Data Curation, Methodology, Software; **Ren J**: Data Curation, Methodology, Software; **Sagayaradj S**: Data Curation, Methodology, Software; **Sapoval N**: Data Curation, Methodology, Software; **Sinner C**: Data Curation, Methodology, Software; **C. Soto D**: Data Curation, Methodology, Software; **Soylev A**: Data Curation, Methodology, Software; **Subramaniyan A**: Data Curation, Methodology, Software; **Syed N**: Data Curation, Methodology, Software; **Tadimeti N**: Data Curation, Methodology, Software; **Tater P**: Data Curation, Methodology, Software; **Vats P**: Data Curation, Methodology, Software; **Vaughn J**: Data Curation, Methodology, Software; **Walker K**: Data Curation, Methodology, Software; **Wang G**: Data Curation, Methodology, Software; **Zeng Q**: Data Curation, Methodology, Software; **Zhang S**: Data Curation, Methodology, Software; **Zhao T**: Data Curation, Methodology, Software; **Kille B**: Data Curation, Methodology, Project Administration, Software, Supervision, Writing – Original Draft Preparation; **Biederstedt E**: Data Curation, Methodology, Project Administration, Software, Supervision, Writing – Original Draft Preparation, Writing – Review & Editing; **Chaisson M**: Data Curation, Methodology, Project Administration, Software, Supervision; **English A**: Data Curation, Methodology, Project Administration, Software, Supervision, Visualization, Writing – Original Draft Preparation; **Kronenberg Z**: Data Curation, Methodology, Project Administration, Software, Supervision, Writing – Original Draft Preparation, Writing – Review & Editing; **J. Treangen T**: Data Curation, Methodology, Project Administration, Software, Supervision; **Hefferon T**: Data Curation, Methodology, Project Administration, Software, Supervision, Visualization, Writing – Original Draft Preparation; **Chin CS**: Data Curation, Methodology, Project Administration, Software, Supervision, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing; **Busby B**: Conceptualization, Data Curation, Funding Acquisition, Methodology, Project Administration, Resources, Software, Supervision, Writing – Original Draft Preparation, Writing – Review & Editing; **J Sedlazeck F**: Conceptualization, Data Curation, Funding Acquisition, Methodology, Project Administration, Software, Supervision, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: Wouter De Coster has received free sequencing consumables and travel reimbursement to speak at conferences from Oxford Nanopore Technologies. Jason Chin and Ben Busby are employees of DNAnexus. Zev Kronenberg is an employee and shareholder of Pacific BioSciences; is a shareholder of Phase Genomics. Christopher Dunn is an employee and shareholder of Pacific BioSciences. FS received travel and conference reimbursements from Oxford Nanopore and PacBio. Qiandong Zeng is an employee of Laboratory Corporation of America Holdings, a company providing clinical diagnostics services.

Grant information: Michael Jochum is supported by NIH/NICHD grant award 1T32HD098068-01A. Wouter De Coster is supported by an FWO junior postdoc grant. Daniel Cameron is supported by NHMRC Ideas Grant 1188098. James Havrilla is supported by R01 GM132713-02S1 for this work. Ann Mc Cartney was supported, in part, by the Intramural Research Program of the National Human Genome Research Institute, National Institutes of Health. Ahmad Al Khleifat is funded by The Motor Neurone Disease Association and NIHR Maudsley Biomedical Research Centre. Timothy Hefferon is supported by the Intramural Research Program of the National Library of Medicine, National Institutes of Health. BK, TT & NS were supported, in part, by a C3.ai Digital Transformation Institute COVID-19 award FS & MH was supported by NIH UM1 HG008898 NIH.
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

How to cite this article: Mc Cartney AM, Mahmoud M, Jochum M *et al.* **An international virtual hackathon to build tools for the analysis of structural variants within species ranging from coronaviruses to vertebrates [version 2; peer review: 1 approved, 3 approved with reservations]** F1000Research 2021, **10**:246 https://doi.org/10.12688/f1000research.51477.2

First published: 26 Mar 2021, **10**:246 https://doi.org/10.12688/f1000research.51477.1

Introduction

Structural variants (SVs) comprise a number of genomic imbalances including copy number variations (CNVs), insertions (INS), deletions (DELs), inversions (INVs), duplications (DUPs), and inter-chromosomal translocations.[1–3] SVs have been implicated as clinically significant mutations with proven associations to multiple diseases.[4,5] Despite next-generation sequencing becoming increasingly common within the field of biomedical research, several practical challenges exist for comprehensively detecting and evaluating SVs particularly in regard to the high false positive or negative rate along with the accuracy of breakpoint prediction.[6,7] While SV detection with genotyping arrays remains the most commonly used method, the toolbox for SV detection is expanding to incorporate the advancements in third generation sequencing technologies provided by Pacific BioSciences,[8] Oxford Nanopore Technologies,[9,10] optical mapping and NanoString[11] to name a few. These advancements offer potential for solving previously unresolved structural variants.

In October 2020, 62 scientists from nine nations worked together remotely in the Second Baylor College of Medicine & DNAnexus hackathon, focusing on different related topics on SV, Pan-genomes, and SARS-CoV-2 related research. Consequently, this international structural variation hackathon meeting focused on eight themes: 1.) efficiently genotyping vast quantities of SVs; 2.) mapping CNVs to SV types; 3.) detecting and validating SVs for SARS-CoV-2; 4.) filtering high-confidence SV calls for clinical genomics; 5.) SV read-based phasing for haplotype analysis; 6.) genome graph generation without a reference; 7.) machine learning approaches to predict lab-of-origin of a sample.; and 8.) gene-centric data browsing for SV analysis.

Overall, this manuscript details our tools' objectives, value-add, implementations, and applications and sets the foundation for further concept development beyond. In this article we present 10 software tools that were the results of this hackathon.

nibSV: efficient genotyping of SVs from short read datasets. Detection of SVs longer than a short-read (<500bp) DNA trace is very challenging as the SV allele becomes split across multiple reads. To this end, long read sequencing technology is preferential for overcoming this challenge however, although long read sequencing has proven more accurate in SV identification, obtaining accurate allele frequencies across a population is important in order to rank and identify potential pathogenic variations.

Thus, it is still important to genotype SV events in pre-existing short read datasets such as those provided by the 1000 genomes project, Topmed, CCDG, etc. Recently, two main approaches, Paragraph[12] and VG,[13] have achieved this with high accuracy even for insertion SV events. However, these methods are computationally expensive particularly when the number of SVs to be genotyped per sample increases. Furthermore, and maybe more crucially, both methods rely on precise breakpoints that do not change in other samples, an assumption that is potentially flawed particularly over repetitive regions. NibbleSV is a software package able to efficiently genotype vast quantities of SVs whilst also using a kmer catalogue of SVs in order to circumvent the need for re-mapping the same dataset to different versions of the same reference genome (e.g. hg19 vs. GRCH38 vs. CHM13), again aiding computational efficiency (Figure 1).

CNV2SV: supplement CNV calling in SV detection. Copy number variations (CNVs) are a subset of structural variants (SVs) consisting of duplications and deletions. CNVs constitute an important part of humangenetic diversity but are also known to be involved in the pathogenesis of multiple diseases, including 15% of breast cancers.[14,15] In clinical NGS applications, CNVs are commonly detected using short-read sequencing. To this end, coverage changes across the reference genome are interpreted by CNV callers as either putative duplication or deletion events. However, short reads grant only limited insight into the larger structural context of the called CNVs. Recent advances in ultra-long read sequencing have enabled projects such as the complete telomere-to-telomerere construction of a haploid human cell line (T2T CHM13).[16] This opens up the opportunity to better understand putative CNVs identified using short reads in the light of the larger, distant or nested structural events as part of which they may arise. To facilitate this, we have developed CNV2SV, a method to automatically link CNV calls to corresponding SVs from a whole-genome alignment. In turn, this may not only aid in improving our understanding of structural changes encompassing large CNVs but also in the resolution of CNVs with complex breakpoints. We demonstrate CNV2SV for a dataset based on the haploid human genome T2T CHM13, using established methods for CNV calling using short reads and SVs called from genome-genome alignment. Agraphical outline of the CNV2SV pipeline is shown in Figure 2.

CoronaSV: SV pipeline for SARS-CoV-2. While deletions have been reported in several SARS-CoV-2 genomes at consensus level,[17,18] the confidence in how these deletions are detected has not yet thoroughly been evaluated. Existing methods for detecting SV at the individual read level often suffer from false positive calls.[19,20] Additionally, analyses with different variant calling pipelines often result in inconsistent calls.[21,22] To examine the landscape and extent of SV across SARS-Cov-2 genomes, a method for generating accurate and trustworthy SV calls is needed. With this in mind, we developed the CoronaSV bioinformatics pipeline.

CoronaSV is a SV detection and SV validation pipeline for SARS-CoV-2 that combines an ensemble SV calling approach that relies on both long read and short read sequencing technologies (Figure 3). Both assembly-based and read based SV detection methods are used by CoronaSV. By combining different sequencing technologies and variant detection approaches, we can identify both a) confident SV call set and b) artifacts that may result from specific technologies + computational approaches.

CleanSV: Filtering High-quality SVs. Short-read sequencing is performed within clinical genomics to both inform and directly guide patient care. This has been immensely successful for various Mendelian disorders, where patients now routinely have their genomes sequenced to detect high-quality variants. Indeed, this approach has been utilized within clinical genomics for close to a decade,[23] often to correct misdiagnoses (see[24] for a recent example).

Within precision oncology, short-read sequencing (normally targeted sequencing or whole exome sequencing (WES)) has proved successful not only illuminating the nature of specific cancers, but also guiding novel drug development. Today routine sequencing is used to apply therapies for specific cancer subtypes, and influence the treatment of individual patients.[25,26] For clinical work, samples from tumors are sequenced for somatic variants in well-studied oncogenes/tumor suppressor genes. Bioinformaticians will then manually investigate the variant calls within IGV in order to validate how accurate they are, and finally send reports summarizing these data to clinicians.

However, SV calling using short-read data is marred by high false positive (FP) call rates, sometimes up to 90% with modern callers.[10,27,28] As a result, manual curation for each patient proves oppressively time-consuming for the needs of modern precision care and is prone to human error. Even though aneuploidy has been long studied for its role in tumor progression (see[29] for a recent review), due to algorithmic uncertainties, routine inclusion of high-confidence SVs within clinical reports is often infeasible today.

Therefore, there exists a pressing need within bioinformatics to develop methods to remove false positives from the outputs commonly used SV callers, and benchmark their performance across a variety of assays (including a range of sequencing depths and tumor purities). Individual SV callers rely upon specific strategies to detect SVs, which makes the nature of the false positives algorithm-specific. Having access to a call set with a lower false positive rate would certainly not eliminate the requirements of manual curation, but it would make the problem more tractable.

The goal of this project was to develop a set of publicly available filters tailored for cancer genomics which have been measured to perform reliably across popular SV callers, as the filters must be specific to the SV caller used. Using a large cohort of high-quality normal whole genome sequencing (WGS) samples, we perform systematic false positive filtering. SVs labeled by the algorithm as somatic have evidence as actually being germline, while others are algorithmic artifacts. With such filters, bioinformaticians would have access to a set of high-quality somatic calls to manually curate, which could finally result in more robust clinical reports.

Sniphles: Phasing SVs with parallel programming. Phasing infers the correct cis or trans relationship between different heterozygous variations facilitating accurate haplotype reconstruction.[30] Protocols and programs utilizing molecular phasing (chromosomal separation at the bench before sequencing), pedigree-based phasing (matching parental and offspring genotypes to understand the haplotype), population-based phasing (using genotype data from large cohorts to infer haplotypes), and read-based phasing (mapping sequencing reads with the same variants to construct a haplotype) are all successful approaches to phasing next-generation sequencing data.[31] The long-reads of third generation sequencing have bolstered our ability to phase longer and more comprehensive haplotype blocks.[32] More comprehensive haplotype blocks increase our ability to accurately phase structural variants.

The goal of this project is to develop a wrapper script around the Sniffles SV caller[10] to properly phase SV and augment the ability of Sniffles to accurately call SV (Figure 5). This result is obtained by using phased reads generated by SNV phasing tools such as WhatsHap or LongShot,[30,33] and subsequently call SVs on the haploid phase blocks separately using temporary files before finally merging both haplotypes to obtain a single VCF file. As this algorithm processes each phase block separately this is attractive for parallelization. Our wrapper script additionally makes Sniphles compatible with alignments in the CRAM format.

Swagg: Structural Variation With Annotated Graph Genomes (Swagg). Most graphical approaches to variant calling only use genome graphs. While this information helps illustrate variation on a genomic level, it does not show variation on the individual protein level. To help leverage the power of graph approaches for SV calling, we introduce a pipeline that delivers both protein and genome graphs.

Swagg is a pipeline that enables the construction of genome graphs from read data (Figure 6). The input into the pipeline is sequence reads with or without a reference genome(s). Reads can be short reads or preprocessed (basecalled) long reads. These reads are then assembled into longer contigs which are mapped back to the reference genome to highlight discrepancies with the reference genome. These discrepancies can be caused by real mutations or sequencing artifacts and easily identified using SV tools, which output VCF files for each read set. These VCF files are taken together to make the genome graph at the end of the pipeline. The overall pipeline and intertwined modules are shown below. In addition to the pipeline for creating graph genome and graph proteins, we also have a module for simulating reads based on an input reference genome.

PanOriginSV: detecting lab-of-origin. The advent of novel synthetic biology methods and organic bench-top synthesis toolkits like CRISPR[34] has enabled rapid developments in genetic engineering. However, this progress has also introduced biosafety concerns surrounding the intentional or unintentional misuse of these tools. In order to increase accountability, the lab-of-origin studies attempt to map a set of plasmids to their lab-of-origin. Subsequently, the Genetic Engineering Attribution Challenge (GEAC) was announced, inviting open source tools from the community that could best predict the lab-of-origin.

Previous methods have employed machine learning or deep learning-based approaches that despite their promise, suffer from sub-optimal accuracy, long training times as well as explainability issues. Recently, a new alignment based tool *PlasmidHawk*[35] reported higher accuracy than machine learning tools. *PlasmidHawk* relies on linear pangenome constructs to align query sequences to a pangenome in order to best determine the Top-1 and Top-10 candidate labs. Though *PlasmidHawk* has a higher accuracy, the runtimes to create the linear plasmid are non-scalable to larger datasets. Another drawback being the linear pangenome doesn't incorporate SV, which could be important to predict hard-to-classify sequences. To address some of these challenges, we propose a tool PanOriginSV that combines machine learning approaches with graphical pangenome based alignment to predict lab-of-origin (Figure 7).

PanOriginSV creates multiple pangenome graphs from similar training sequences using BCALM[36] creating a variation graph that incorporates SV and aligns the sequences back to the graph using GraphAligner.[37] The most similar training sequences for graph construction are clustered using MMSEQ2.[36,38] After this, top alignments, scores to the pangenome and sequence metadata are considered as features for a downstream machine learning model towards lab-of-origin prediction.

GeneVar: SV Browser. Next-generation sequencing provides the ability to sequence extended genomic regions or a whole-genome relatively cheaply and rapidly, making it a powerful technique to uncover the genetic architecture of diseases. However, significant challenges remain, including interpreting and prioritizing the identified variants and setting up the appropriate analysis pipeline to cover the necessary spectrum of genetic factors, which includes expansions, repeats, insertions/deletions (indels), SV and point mutations. For those outside the immediate field of genetics, a group that includes researchers, hospital staff, general practitioners, and increasingly, patients who have paid to have their genome sequenced privately, the interpretation of findings is particularly challenging. Although various tools are available to predict the pathogenicity of a protein-changing variant, they do not always agree, further compounding the problem. Furthermore, with the increasing availability of next-generation sequencing data, non-specialists, including health care professionals and patients, are obtaining their genomic information without a corresponding ability to analyse and interpret it as the relevance of novel or existing variants in genes is not always apparent. Similarly SV analysis[39,40] and its interpretation requires care in regard to sample and platform selection, quality control, statistical analysis, results prioritisation, and replication strategy.

Here we present GeneVar, an open access, gene centric data browser for SV analysis (Figure 8). GeneVar takes as input a gene name or ID and produces a report that informs the user of all SVs overlapping the gene and any non-coding regulatory elements affecting expression of the gene. The tool is intended to have a clinical focus, informing the interpretation of SV pertaining to a gene name provided by the user.

SVTeaser: simulated data for SV benchmarking. SV detection tools often have a large number of wrongly detected variations[19,27] requiring benchmarking to assess method quality before finalizing a workflow. Few tools are currently available to simulate data for SV benchmarking. SVTeaser is a tool for rapid assessment of SV calling fidelity with two main use-cases: 1) genotyping a set of pre-ascertained SVs and 2) benchmarking a new algorithm against pre-existing tools across a range of parameters. Users simply supply SVTeaser with a reference sequence file (.fasta) and, optionally, a set of SVs (.vcf). SVTeaser outputs simulated reads across a range of read lengths and depths and provides a downstream dataframe based analysis framework for evaluating accuracy (Figure 9). SVTeaser achieves rapid assessment by downsampling the full reference to a subset of numerous 10kb samples to which it adds SVs.

XSVLen: haplotype-resolved assemblies for benchmarking SVs. Since the development of a "gold standard" SV set, sequencing technologies and assembly algorithms have improved to enable nearly complete haplotype-resolved assemblies of human genomes. XSVLen is a framework (Figure 10) to use haplotype-resolved assemblies for benchmarking SV detection algorithms. Each variant call may be considered an operation to be applied to the reference genome. Our framework for benchmarking SV callsets is to apply SV operations to the reference genome and compare the modified reference against the haplotype-resolved assemblies. This approach allows for SV calls that are different but produce similar sequences due to the repetitive nature of the genome to be scored as valid. In this manner, all variants in a region that is accurately assembled in both haplotypes may be benchmarked using this approach. We demonstrate the effectiveness of this approach by scoring SV calls generated from Oxford Nanopore reads on the HG002 genome[41] using CuteSV[42] and comparing against gold-standard calls by Truvari (https://github.com/spiralgenetics/truvari). This approach can be extended to use any haplotype-resolved assembly to benchmark SV callsets in additional genomes, enabling benchmarks as a distribution across call sets.

Methods

Implementation

nibSV: NibbleSV is a lightweight, scalable and easy to apply method to identify the frequency of SV events across short read data sets. As such, nibSV extracts kmers that are informative if an SV is present or if an SV is absent given the breakpoints of the previous predicted SV. Subsequently, nibSV scans the short read bam or fastq file for the presence of these k-mers and counts their number of occurrences. In the end, nibSV extends the VCF file with tags holding information about the number of times an SV is supported by kmers or not (Figure 1).

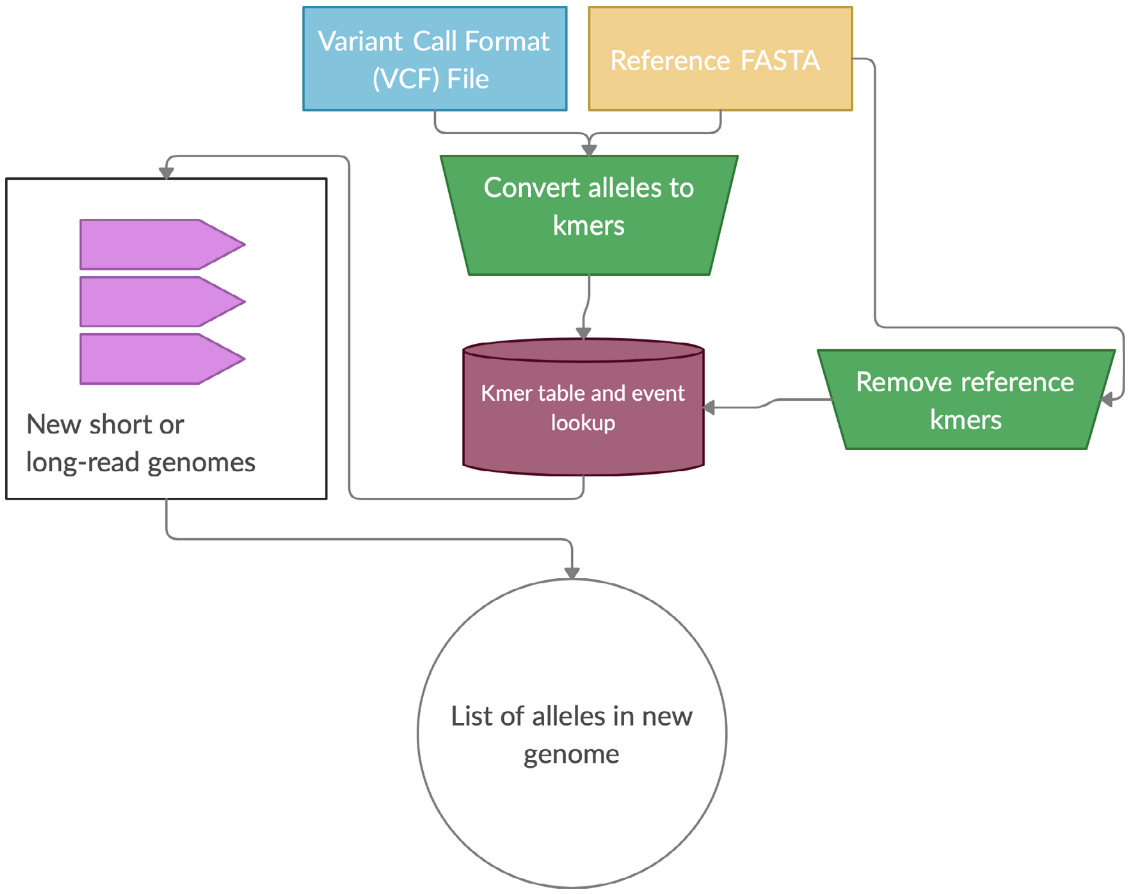

Variant Call Format (VCF) File
Reference FASTA
Convert alleles to kmers
New short or long-read genomes
Kmer table and event lookup
Remove reference kmers
List of alleles in new genome

Figure 1. Overview of the NibbleSV software package workflow. Overview of NibbleSV workflow that utilises an input reference genome and file containing variant calls to generate a list of genotyped alleles using a kmer based strategy.

CNV2SV: CNV2SV is a tool for connecting CNVs identified using short-read sequencing to structural variations (SVs) from a whole-genome alignment (Figure 2). As input, CNV2SV requires two VCF files containing the CNV and SV calls to be linked, respectively, as well as the reference genome sequence for which CNVs were called (.fasta). For each CNV, we identify matching SVs representing putative tandem duplication or translocation events. After validating each putative link through pairwise sequence alignment, the details for each CNV-SV match are saved in a .tsv file along

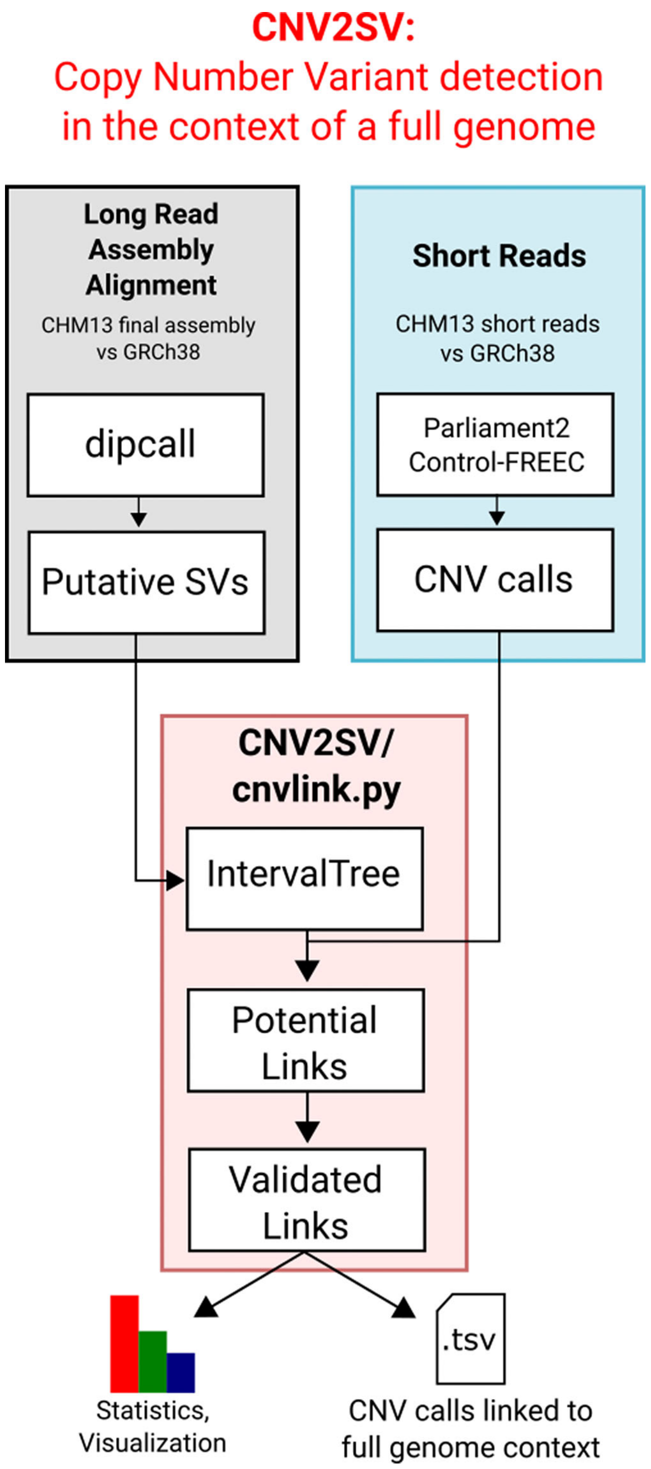

CNV2SV:
Copy Number Variant detection in the context of a full genome
Long Read Assembly Alignment
CHM13 final assembly vs GRCh38
dipcall
Putative SVs
Short Reads
CHM13 short reads vs GRCh38
Parliament2 Control-FREEC
CNV calls
CNV2SV/ cnvlink.py
IntervalTree
Potential Links
Validated Links
.tsv
Statistics, Visualization
CNV calls linked to full genome context

Figure 2. A graphical overview of the CNV2SV pipeline. CNV2SV software pipeline that utilises both short and long read data as input to calculate the frequency of copy number variants across complete genomes.

with summary statistics. The output can then be further visualized using a circular plot showing the genomic positions for each discovered link. The results of CNV2SV may be interpreted in order to better understand the structural changes underlying individual CNV calls as well as evaluating and potentially improving the accuracy of break point detection for structural events of different sizes and complexities.

CoronaSV: CoronaSV is a method developed for generating accurate and trustworthy SV calls across SARS-Cov-2 genomes. The tool utilises available SRA data from SARS-CoV-2 isolates that have been sequenced with both Illumina and ONT platforms. CoronaSV utilizes a combination of three different approaches: read-based SV detection with paired-end Illumina reads and ONT long-reads, as well as assembly-based SV detection using both short and long-reads (Figure 3). All the software packages used by CoronaSV can be installed via the Conda package manager (https://github.com/conda/conda).

CleanSV: The goal of the hackathon project was to develop filters and QC checks to remove false positive calls from common SV callers. Currently, within clinical genomics, it's exceptionally difficult to categorize true positives from false positives, thus making accurate diagnoses virtually impossible. The situation is even more complicated within clinical oncology, as researchers need to precisely separate true somatic calls from false positives and (potential) germline calls. In order to aid with precision SV calling, the team wrote a set of scripts to be used with short-read SV callers, allowing researchers to better generate a set of high-quality SVs to further investigate manually (Figure 4).

For cancer genomics, groups normally develop in-house filters to improve the precision of SV calling. The scripts developed for this project accept as input GRIDSS,[43] Manta,[44] DELLY,[45] and SvABA[46] calls from short-read WGS data,

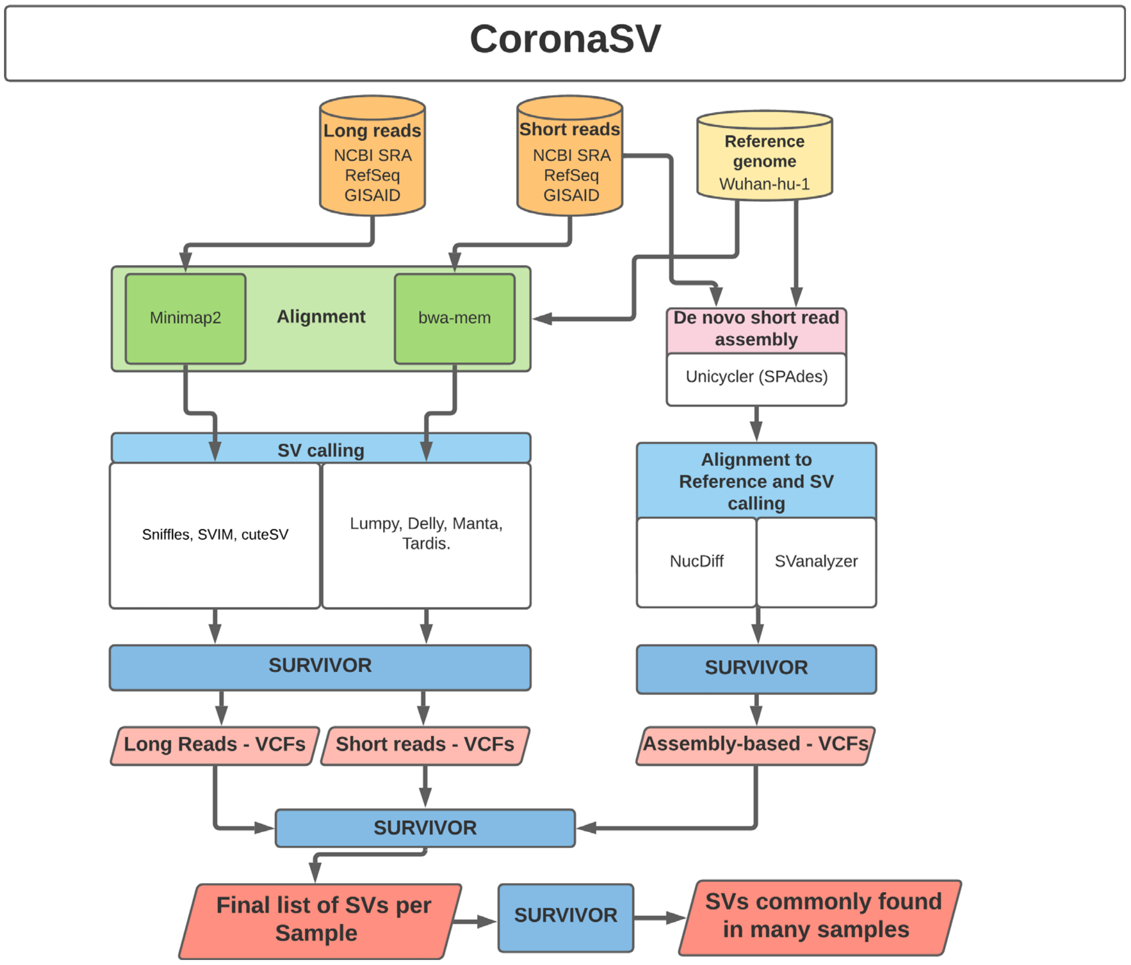


Figure 3. Illustration of CoronaSV software implementation. Illustration of the CoronaSV package workflow that takes SARS-CoV-2 short and long read data types along with a SARS-CoV-2 reference genome as input and generates a set of commonly found SVs.

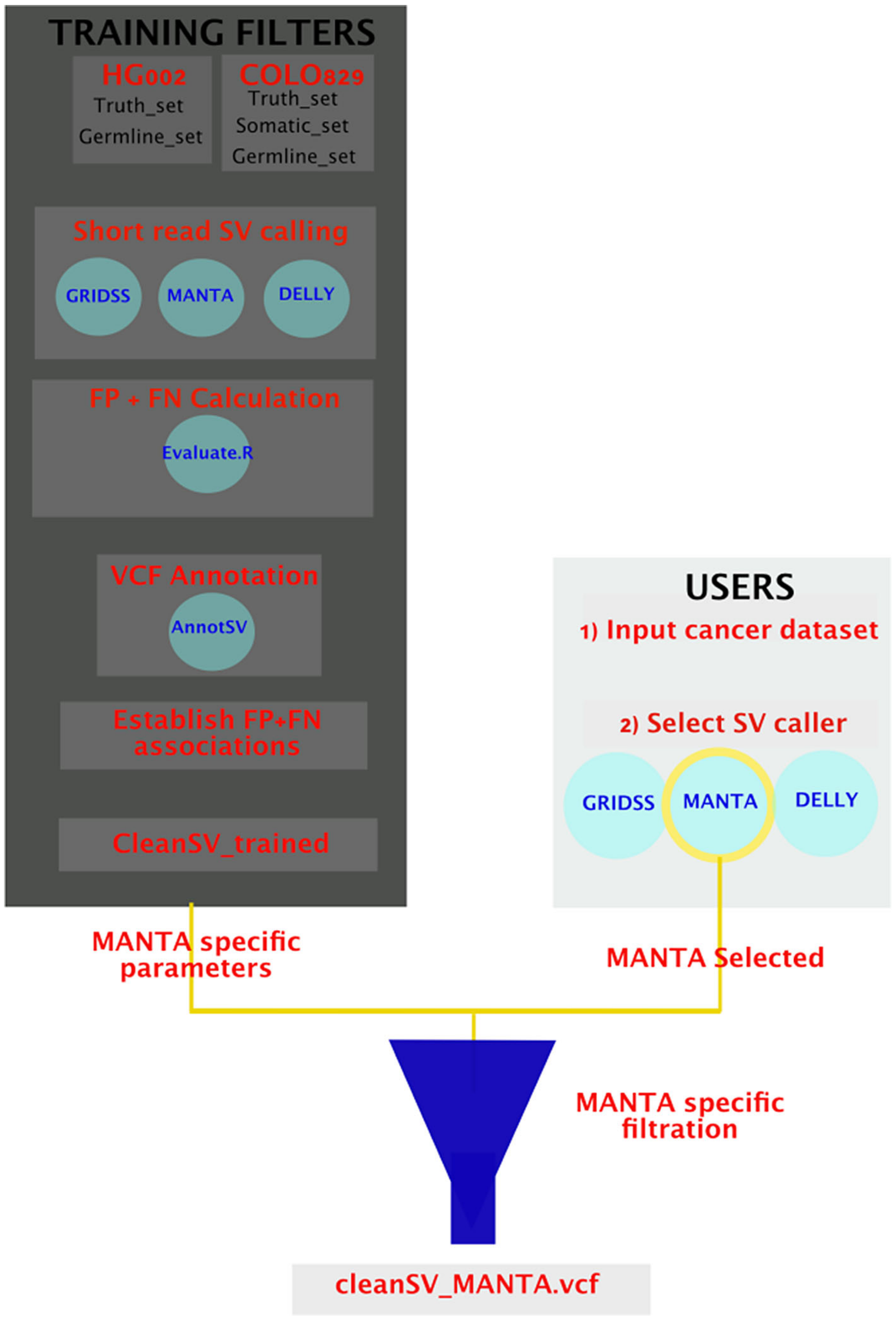


Figure 4. An illustration of the approach used by CleanSV to generate and implement filtration of SV calls. CleanSV pipeline highlighting methods used to generate adequate filters that can be utilised by clinicians to filter false positive and mislabeled SV calls from short-read cancer datasets.

along with a manually-curated reference set of calls designated as ground truth. The reference set was curated using a paired melanoma and normal lymphoblastoid COLO829 cell lines using four different technologies (Illumina HiSeq, Oxford Nanopore, Pacific Biosciences and 10x Genomics), along with extensive external validation.[47] Using the reference set, we proceeded to investigate the presence and nature of false positives from the initial callsets. (Note that we focused on WGS for this hackathon, but a similar approach could be applied to other assays such as WES.) Samplot[48] visualization of read data allowed manual curation of parameters associated with FP calls, and associations between AnnotSV[48,49] annotated parameters and FPR helped identify additional FP-associated SV parameters (Figure 4). Along with the manually-curated reference set, the panel of normal (PON) used for further filtering was generated from a compiled set of high-quality germline calls using 3,782 normal samples freshly-sequenced at a median depth of 38x by the Hartwig Medical Foundation.[50,51]

Sniphles: The main idea is to phase the identified SVs. We use two approaches; the first is to extract the tagged reads from the bam file and use these reads to phase the SVs if not conflicted. The second approach is to split the haplotype bam file based on the haplotype tag, using each split bam file to call SVs separately, this method called (Sniphles). Sniphles utilize information impeded in haplotypes, bam file and reads info support SVs. This method phases structural variants and augments the ability of Sniphles to accurately call SVs (Figure 5). Sniphles is implemented in Python 3, and it takes a haplotyped bam, and a SV VCF file as input and produces a phased VCF file as output.

Swagg: Structural Variation with Annotated Graph Genomes (SWAGG) is a pipeline to make genome graphs from read data. The input into the pipeline is reads either with or without reference genome(s). Reads can be short-reads or preprocessed (basecalled) long-reads. Reads are assembled into longer contigs, and contigs are mapped back to the reference genome to look for discrepancies with the reference genome. These discrepancies can be either real mutations or sequencing artifacts, and are found using structural variant tools which output VCF files for each read set. These VCF files are taken together to make the genome graph at the end of the pipeline (Figure 6).

PanOriginSV: This tool is a lab-of-origin prediction tool that combines machine learning approaches with graphical pangenome based alignment to predict lab-of-origin (Figure 7). PanOriginSV is implemented in Python 3 and uses the scikit-learn package for deploying machine learning models. PanOriginSV also relies on MMseqs2 for clustering, BCALM for graph construction and minigraph for graph alignment. Given a training set of engineered plasmids and their source labs, this software can predict the lab of origin of a test set of plasmids.

GeneVar: The GeneVar tool was developed to help inform the clinical interpretation of structural variants pertaining to a user-provided gene. This software is an open access, gene-centric data browser for SV analysis. GeneVar is a web page

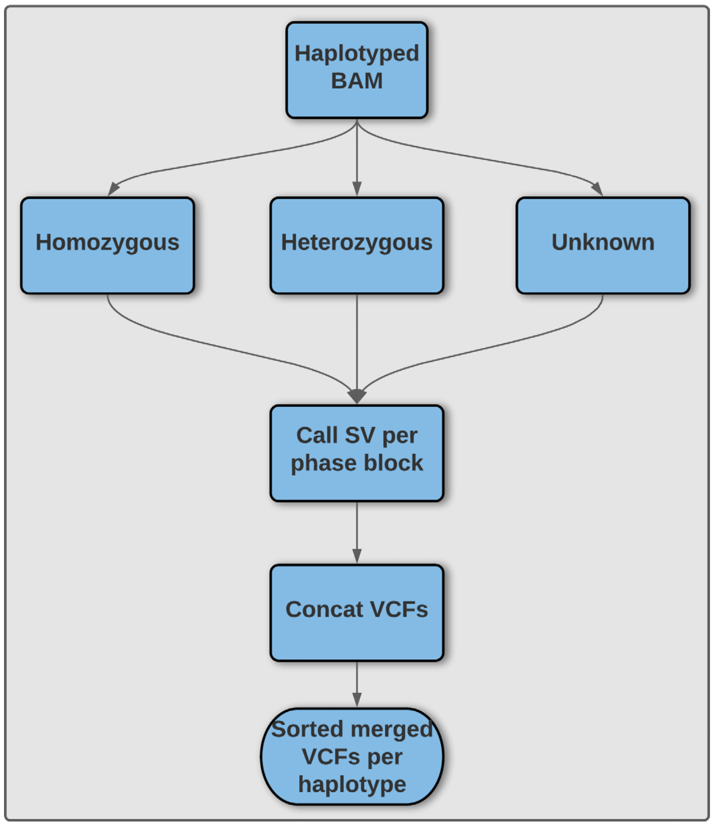


Figure 5. Illustration of methodology utilized by Sniphles to produce a phased structural variant call set. **An overview of the Sniphles pipeline demonstrating how a haplotyped input bam file is used to generate a phased structural variant call set.**

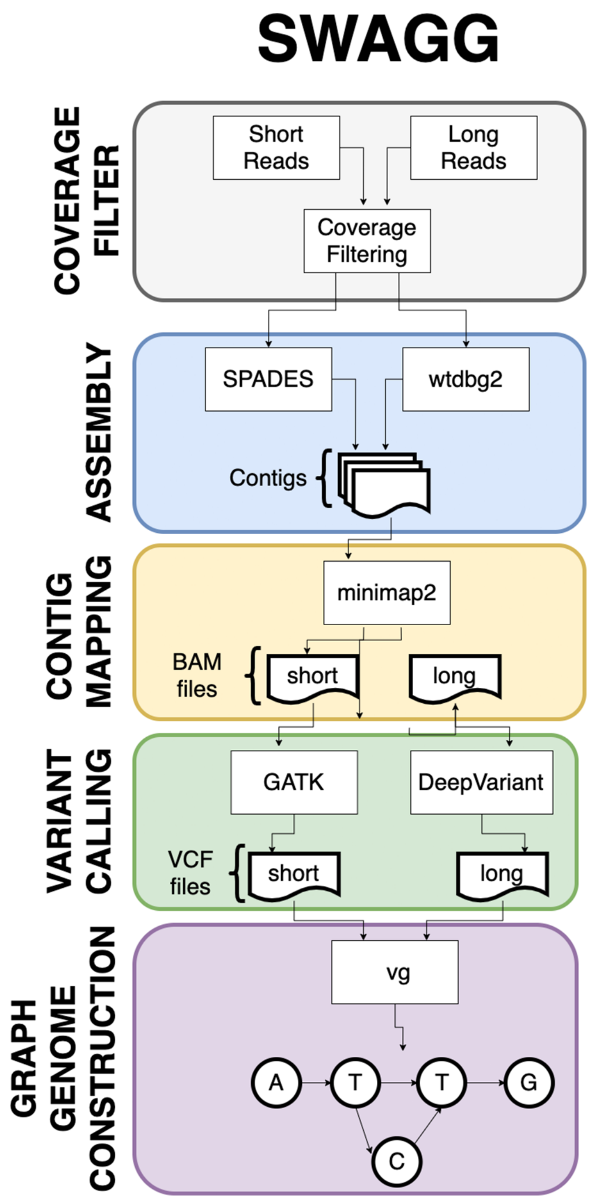


Figure 6. Outline of the SWAGG software package workflow. Illustration of the SWAGG pipeline that utilises both long and short read datasets for the construction of graph genomes.

application (Figure 8). After entering the gene name (HGNC, Ensembl gene (ENSG), or transcript (ENST) identifier) in the search box on the homepage, the user is directed to the summary of the gene-specific page. GeneVar is available on GitHub (https://github.com/collaborativebioinformatics/GeneVar). The repository provides detailed instructions for tool usage and installation. A bash script for automated installation of the required dependencies is also provided.

SVTeaser: SVTeaser is a tool for rapid assessment of SV call fidelity created for geneticists designing experiments to genotype a set of pre-ascertained SVs and bioinformaticians benchmarking a new algorithm against pre-existing tools across a range of parameters (Figure 9). Users are required to supply SVTeaser with a reference sequence file (.fasta) and, optionally, a set of SVs (.vcf). SVTeaser outputs assorted statistical metrics across a range of read lengths and depths. SVTeaser achieves rapid assessment by downsampling the full reference to a subset of numerous 10kb samples to which it adds SVs.

XSVLen: This software is a framework for benchmarking SV detection algorithms against haplotype-resolved assemblies in which variants are validated by comparing sequence content rather than comparing breakpoints of variants (Figure 10). XSVLen validates a VCF file with sequence-resolved variants including those produced by cuteSV or

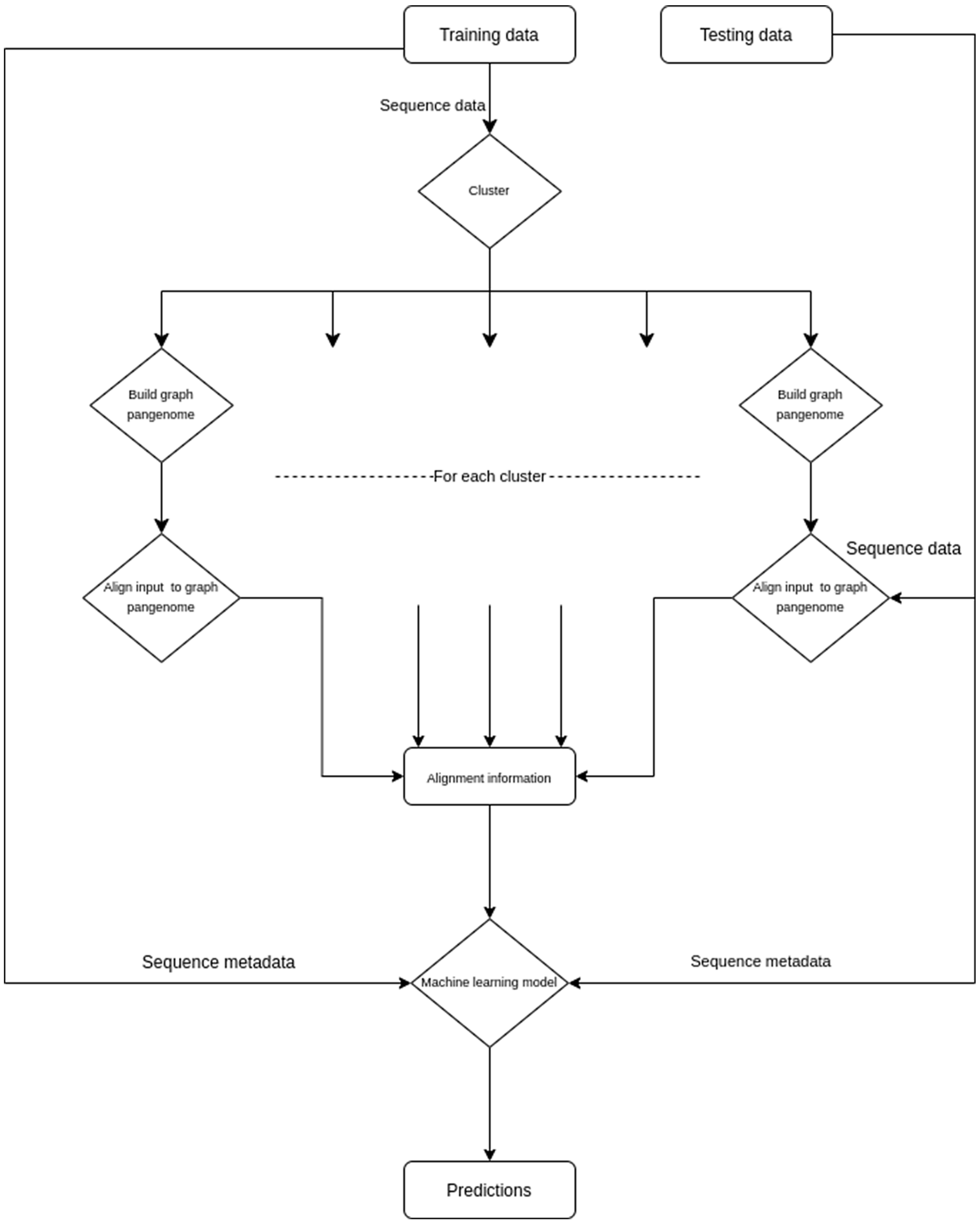


Figure 7. Implementation strategy of the PanOriginSV software package. Pipeline demonstrating the PanOriginSV software package that's implementation determines that lab-of-origin input sequencing data.

Sniffles 10) by creating a test sequence for each variant defined by extracting the region around a variant, and applying the SV operation (insertion or deletion). The test sequences are mapped to the haplotype-resolved assemblies, and high-identity alignments validate calls. All methods are open-source licensed and have been made available on GitHub: https://github.com/collaborativebioinformatics.

Operation

nibSV: nibSV requires a reference genome and VCF file that includes all the SV that should be genotyped (Figure 1). Next, allele kmers for the reference and alternative are extracted. The extraction process includes each site's flanking

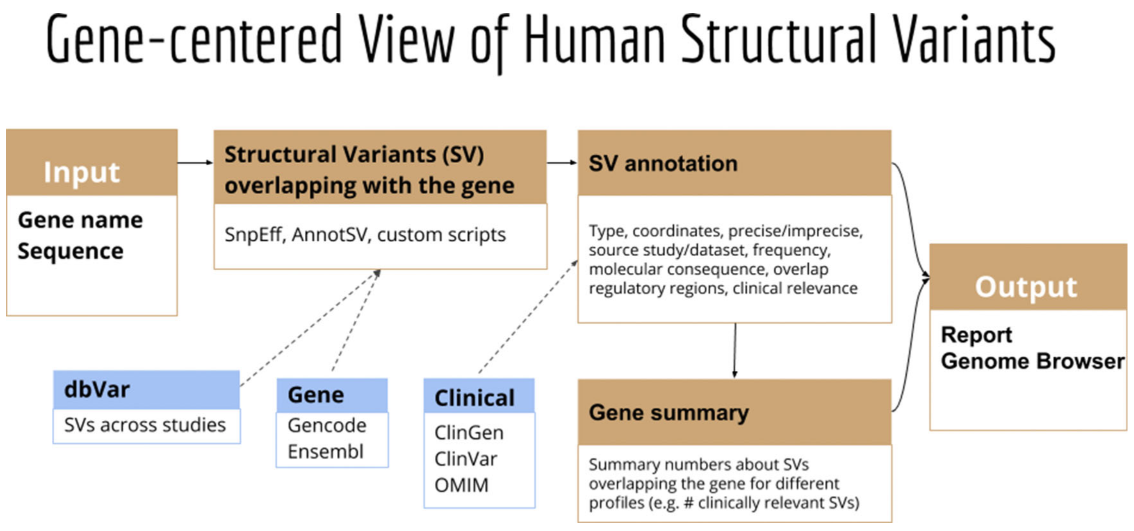

Gene-centered View of Human Structural Variants
Input
Gene name
Sequence
Structural Variants (SV) overlapping with the gene
SnpEff, AnnotSV, custom scripts
SV annotation
Type, coordinates, precise/imprecise, source study/dataset, frequency, molecular consequence, overlap regulatory regions, clinical relevance
Output
Report
Genome Browser
dbVar
SVs across studies
Gene
Gencode
Ensembl
Clinical
ClinGen
ClinVar
OMIM
Gene summary
Summary numbers about SVs overlapping the gene for different profiles (e.g. # clinically relevant SVs)

Figure 8. Methodology used by GeneVar for the production of summary report made available through genome browser. A graphical representation of the pipeline used by GeneVar in order to provide clinicians with a comprehensive summary of SVs associated with a user provided gene name.

regions. Subsequently, the occurrence of these k-mers in the reference fasta file are counted. This step is necessary to prevent k-mer miscounting between reference vs. alternative allele. To enable scaling of nibSV for large data sets, the results of these two steps are written into a temporary file, which is all that is needed for the actual genotyping step. During the genotyping step, nibSV uses this small temporary file and the bam/cram file of the sample and identifies the presence/absence of the reference and alternative k-mer across the entire sample. This is very fast and requires only minimal resources of memory as the number of k-mers is limited. On completion of nibSV a scanning of the bam/cram file is carried out reporting which SV have been re-identified by adding a tag in the output VCF file of this sample (Figure 1). The VCF per sample can then be merged to obtain population frequencies. The VCF per sample can then be merged to obtain population frequencies. nibSV requires 4Gb of memory, a single core and around 2GB hard disk space to store its index from e.g. GIAB HG002.

CNV2SV: To link copy number variant calls to matching structural variants, CNV2SV requires three input files: The VCF file containing the CNVs (typically called from short reads), the VCF file with the structural variants (typically called from a genome-genome alignment) as well as the reference sequence(.fasta) that the CNVs were called for. CNVs are then linked to matching SVs in two steps: First, individual CNVs are queried against an interval tree structure containing all SVs from the genome-genome alignment to find adjacent CNV-SV pairs (<1000 bp apart, putative tandem duplication events). For CNVsthat had no match identified in this way, the search is successively extended to the whole genome (putative translocation events). Next, all putative CNV-SV links are evaluated by pairwise sequence alignment using mappy (Python binding for minimap252), with a standard sequence identity threshold of 0.8. The main output comprises detailed information about the discovered links for each CNV. This includes the number of best matched SV, its genomic coordinates and CNV-SV sequence alignment as well as summary statistics useful for evaluating e.g. the quality/resolution of breakpoints identified by the CNV callers. Furthermore, all additionally identified adjacent and distant SVs are reported separately for each CNV. The raw output can be further visualized toshow the CNV-SV links identified across the genome in a circular plot, as well as summary statistics for the linking results. For the results presented in theuse cases section, we applied CNVnator53 and dip call to generate CNV and SV calls for the T2T CHM13 data set and GRCh38, respectively. A quick-start example for the CHM13 and GRCh38 data is available on our GitHub page. In addition, we hosted a detailed description of the output data on the GitHub page. The output data can be further visualized using the additionally provided scripts, which include circular plots to display the positions of each discovered CNV-SV link. System requirements (see GitHub for more information): CNV2SV has been tested to work on a desktop system on the CHM13 data set with an Intel® i7-6700K Processor (4.00 Ghzquad-core), 32GB RAM (less may be required), 50GB free disk space and running Unix-like operating system (e.g. Ubuntu-based distribution) or Windows subsystem for Linux running Ubuntu. The initial genome-genome alignment (CHM13vs GRCh38) was computed on a cloud-based platform (DNANexus). CNV2SV requires Python (3.8 or newer) and prerequisite Python packages include interval tree, mappy and pyfaidx. For visualization of the discovered CNV-SV links in acircular plot, R and the R package circlize are additionally required. A full list of package dependencies is available on the GitHub page.

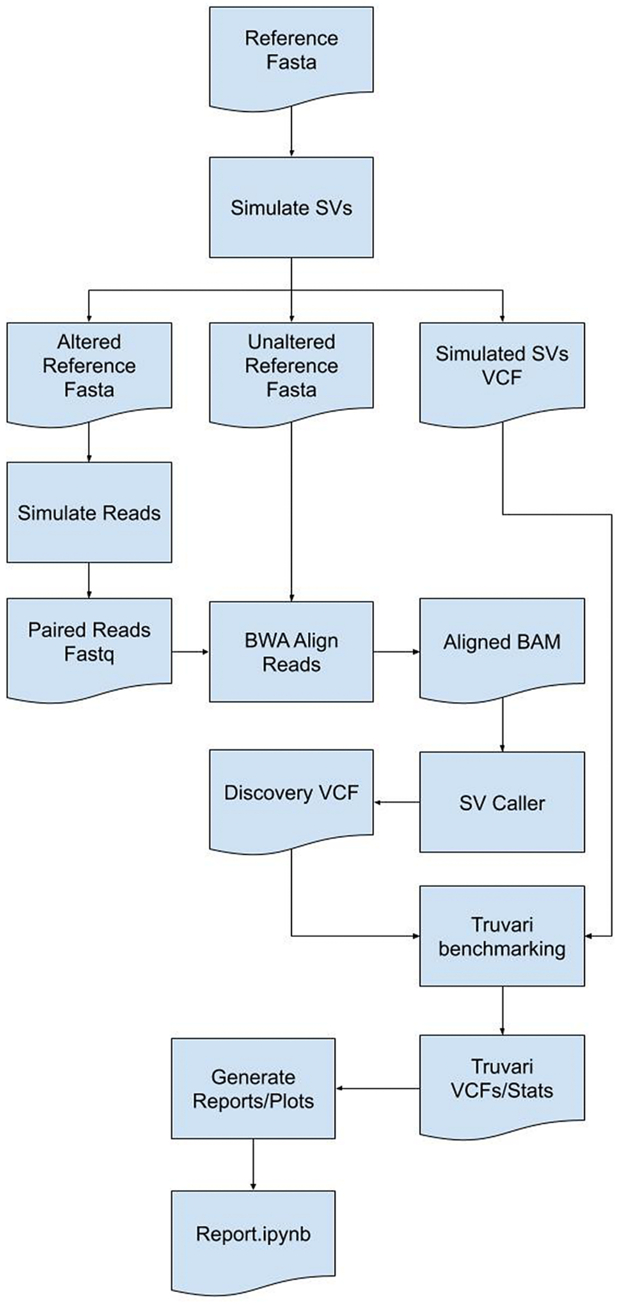


Figure 9. An illustration of the methodology implemented by the SVTeaser software package. SVTeaser software implementation showing how by using a reference genome a set of SVs can be simulated to benchmark SV calling tools to inform experimental design decisions prior to analyses.

System requirements (see GitHub for more information): CNV2SV has been tested to work on a desktop system on the CHM13 data set with an Intel® i7-6700K Processor (4.00Ghz quad-core), 32GB RAM (less may be required), 50GB free disk space and running Unix-like operating system (e.g. Ubuntu-based distribution) or Windows subsystem for Linux running Ubuntu. The initial genome-genome alignment (CHM13 vs GRCh38) was computed on a cloud-based platform (DNANexus). CNV2SV requires Python (3.8 or newer). A full list of package dependencies is available on the GitHub page.

CoronaSV: All software packages used by CoronaSV can be installed via the Conda package manager. Additionally, the CoronaSV workflow is defined using Snakemake. Running the CoronaSV.smk snakemake pipeline handles

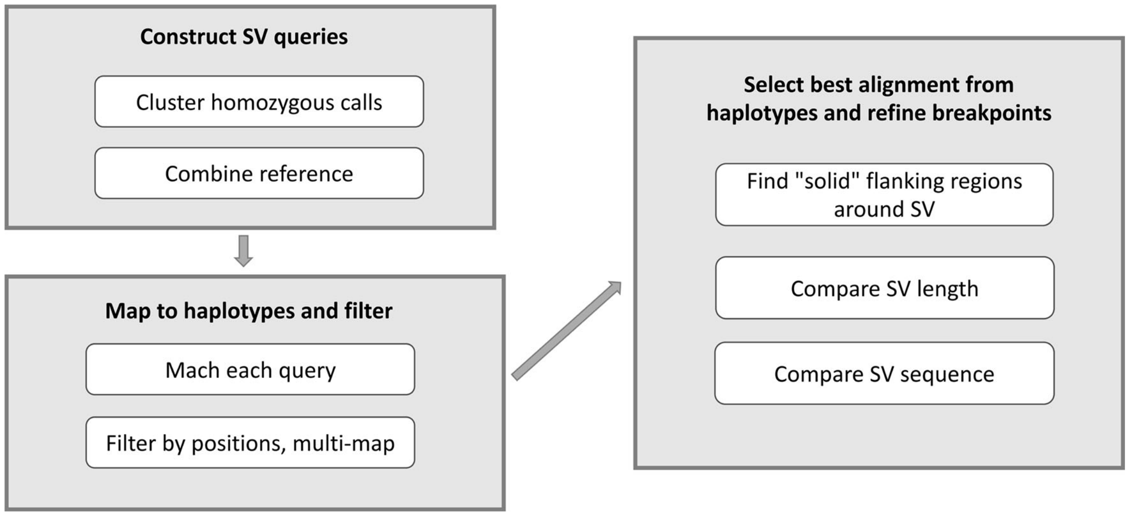

Construct SV queries
Cluster homozygous calls
Combine reference
Map to haplotypes and filter
Mach each query
Filter by positions, multi-map
Select best alignment from haplotypes and refine breakpoints
Find "solid" flanking regions around SV
Compare SV length
Compare SV sequence

Figure 10. Implementation pipeline of XSVLen software package. A graphical representation of the XSVLen software pipeline showing the utilisation of haplotype-resolved de novo assembly and a VCF file to benchmarking SV detection algorithms.

downloading all specified data and processing of sequencing data to variant calls. Each step of the CoronaSV pipeline (Figure 3) has a defined conda environment with exact versions of software specified for easy installation. CoronaSV utilizes three approaches that includes 1) read-based SV detection with paired-end Illumina reads, 2) ONT long-reads and 3) assembly-based SV detection. Illumina paired-end short-reads are trimmed using trimmomatic[52] and mapped to the SARS-CoV-2 reference using bwa mem.[53] After mapping, PCR duplicates are removed with Picard MarkDuplicates (http://broadinstitute.github.io/picard). Structural variants are identified then using Delly,[45] Manta,[44] Lumpy,[54] and Tardis.[55] Nanopore long-reads are filtered using Nanofilt and mapped to SARS-CoV-2 reference using minimap2 with default parameters. SVs are then called using Sniffles, SVIM, and CuteSV. Read quality assessment is carried out by NanoPlot. In order to integrate assembly based methods, de novo SARS-CoV-2 assemblies were generated using Unicycler for short-read sequencing. NucDiff and SVanalyzer tools are used for assembly-to-assembly comparisons. Followup comparative analyses across callsets is implemented by SURVIVOR[56] (Figure 3).

System requirements: CoronaSV is tested on Linux-based systems with multiple illumina and nanopore sequencing data (see GitHub for full list of the testing data). The RAM usage of CoronaSV depends on the size of input data. Peak RAM usage appears during de novo assembly using Unicycler, and 16 GB of RAM is sufficient for the pipeline to run on 8 CPU cores with additional 50Gb of disk space. CoronaSV requires python (version 3.6 or newer) and snakemake. Required tools and package dependencies can be found on GitHub page.

CleanSV: The methods adopted to construct the CleanSVs filtration protocols are shown in Figure 4. In order to generate the data required to develop adequate filters for the application within clinical ontology, structural variants (SVs) were called on Illumina short reads using novoalign hs37d5 HG002 BAM (ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/phase2_reference_assembly_sequence/hs37d5.fa.gz) and Illumina short-read HiSeqX Ten hg19 COLO829 BAM (https://nextcloud.hartwigmedicalfoundation.nl/s/LTiKTd8XxBqwaiC?path=%2FHMFTools-Resources%2FGRIDSS-Purple-Linx-Docker) using GRIDSS,[43] Delly,[45] and Manta.[44] 44 (Note we chose HG00241, the son of the"Ashkenazim Trio", as it has been extensively characterized to serve as areliable reference in human genomics). The set of curated SVs based on HG002 [Cite: https://www.nature.com/articles/s41587-020-0538-8] was used to determine the false positive (FP) SV calls in the short read dataset. Any SV calls that were found outside of the truthset Tier1 bed regions were then filtered. In the sample COLO829, the SV truthset was used to determine the FP SV calls in the short-read dataset. Calls were inspected through manual curation by Samplot.[57] Generated samplots were annotated with UCSC table browser repeat tracks and converted using vcfanno as well as GC content calculated by BEDOPS file conversion with the function.

We then compiled a set of high-quality germline calls using 3,782 normal samples freshly-sequenced at a median depth of 38x by the Hartwig Medical Foundation.[50,51] We intiial hypothesized that such a large cohort could be used to both perform systematic FP filtering and possibly detect calls simply incorrectly labeled as somatic. The calls were filtered if a match within 2bp of the breakpoint was found in the PON. System requirements: The scripts to filter SV calls with either VCF or BEDPE format require R version 3.6.0 or higher, which is available for Linux, Mac OS, and Windows. The analyses within were run on R version 4.0.3, with Biocondcutor version 3.12. For running SV callers, it is recommended to use a HPC environment on Linux.

Sniphles: The Sniphles workflow (Figure 5) requires the following dependencies: Python >= 3.6, Pysam (Version 0.16.0) (https://github.com/pysam-developers/pysam), Cyvcf2 (Version 0.30.2),[58] Sniffles (Version 1.12),[10,42] SURVIVOR (Version 1.0.7),[56] Mosdepth (Version **0.2.6**),[59] Bcftools (Version 1.9),[60] tabix (Version 1.8).[61] The workflow partitions reads in the bam file into groups based on phase blocks and phase status, which enables parallel analysis of the data. The read coverage at each block and each phase is computed with Mosdepth and used to estimate the parameters for calling SVs by Sniffles. Next, the identified SVs per haplotype are concatenated using bcftools. SVs of two haplotypes are combined using SURVIVOR with option "1000 1 0 0 0 0" to merge SVs within 1 kbp between each other and to allow for different types of variants to be considered on different haplotypes. SVs are then force called with Sniffles using this combined vcf file. Force called SVs from each haplotype are combined with SVs of unphased regions as the final output (Figure 5). To facilitate workflow testing, Princess (https://github.com/MeHelmy/princess) was used to align, detect and phase SNVs and SVs from PacBio HiFi reads. The produced Bam from the previous step is the input for Sniphles, where pysam was used for alignment.

Swagg: The minimal system requirements for SWAGG are 8Gb RAM, 1 CPU and 10Gb of storage. Figure 6 demonstrates the implementation and operation of the SWAGG software package. Protein graphs are generated using a multiple sequence alignment of the proteins, then using the tool msa_to_gfa (https://github.com/fawaz-dabbaghieh/-msa_to_gfa) after which this multiple sequence alignment is converted into a graph file in GFA format, with the original sequences embedded as paths in the graph for visualization. This tool is tested with Python 3 and does not require any extra libraries or dependencies. New sequences can be aligned to these graphs using Partial Order Alignment algorithm for example.

PanOriginSV: PanOriginSV has three additional open source dependencies which are MMseqs2 (for clustering), BCALM (pangenome creation) and GraphAligner (for sequence-to-graph alignment). MMseqs2 is the most memory intensive step, and MMseqs2 requires roughly 1 byte per sequence residue. An overview of the pipeline utilised by this tool is highlighted in Figure 7. PanOriginSV performs lab-of-origin prediction in three distinct steps. Firstly, during the training phase PanOriginSV clusters similar sequences using MMseqs2. Further, the most similar clusters having a predefined number of representative lab labels are selected for the pangenome creation. Second, in order to incorporate SV information into the pangenome a graphical pangenome is created using BCALM for each of the clusters identified in the first step. These pangenomes reflect sequence-level structural variation that reveal important differences in highly similar sequences that could belong to different labs thereby reducing the possibility of false positives. The training sequences are then mapped back to their corresponding pangenome graph to obtain important alignment information including but not limited to number of hits and percentage identity of alignment. These are then collated and embedded into a feature vector that is passed on to a machine learning model for prediction.

Thirdly, features from the alignment steps are combined with sequence metadata and input into a random forest classifier that is trained to predict lab-of-origin in a multiclass classification task. Both the top-1 and top-10 predictions are output and compared to previous literature and Genetic Engineering Attribution Challenge (GEAC) benchmarks.

GeneVar: Figure 8 shows the different components of GeneVar, which is a web browser application. The webpage, including data storage, requires only one core with 1 Gb RAM and requires less than 1 Gb of storage. After entering the gene name (HGNC, Ensembl gene (ENSG), or transcript (ENST) identifier) in the search box on the homepage, you will be directed to the gene-specific page containing: 1) Gene-level summary with number of SVs, number of clinical SVs or SVs overlapping clinical SNVs, 2) Links to the gene's page on OMIM, GTEx, gnomAD, 3) A dynamic table with the annotated variants overlapping the gene, 4) A graph with the distribution of the allele frequency for variants matched with gnomAD-SV (50% reciprocal overlap). The profile of the SV to consider, such as type and size range, can be specified on the side bar. Each column in the dynamic table can be "searched" into or reordered dynamically. All data used by the app will be available for download in tab-delimited files. By default, allele frequency is reported based on dbVar[62] and gnomAD genomes and exomes. Furthermore, GeneVar utilises dbVar database and links SV to genes and annotate gene impact, allele frequency, and the overlap with clinically-relevant SVs, SNVs and indels. All data, are available for download in a tab-delimited file. Each variant has been extensively annotated and aggregated in a customizable table. GeneVar is available on GitHub (https://github.com/collaborativebioinformatics/GeneVar). The repository provides detailed instructions for tool usage.

SVTeaser: SVTeaser generates regions from a user provided reference and adds in a structural variant into each region using one of two methods - 1) a call to SURVIVOR simSV[56] which generates random, simulated SVs by introducing variation (deletions (DEL) and insertions (INS) type of SV breakpoints) in DNA sequences, or 2) automatic spike-in of a known SV from an input SV VCF file. Resultant altered reference sequences are then used for Illumina short-read simulation using ART.[63] Parameters controlling simulated sequencing read-length, insert-size, and depth parameters can be altered. Simulated reads can then be mapped to the original, unaltered reference with any mapper of choice; here, BWA was used. Resultant BAM files can then be used to detect SVs using any mapping-based SV caller of choice; here, Parliament2[64] was used to generate calls with manta, breakseq,[60] cnvnator, and lumpy. The resultant VCFs are then matched to the simulated SVs' VCFs using Truvari and output is parsed into a pandas dataframe for report generation. SVTeaser requires installation of Python 3.7, Truvari, SURVIVOR,[56] Vcftools and ART read simulator. The components of SVTeaser are shown in Figure 9.

XSVLen: The XSVLen workflow requires Python3, Minimap2, Nextflow, and R >=3.5.0. As demonstrated in Figure 10, XSVLen takes as input a haplotype-resolved *de novo* assembly, and a VCF file (generated by cuteSV[42] or sniffles[10]) of variants including only insertion and deletion calls. For each insertion or deletion call within the vcf file, a modified reference genome is generated. This modified reference will contain a 1.5kb flanking sequence that either has the sequence removed if a deletion call, or the alternate sequence added between the flanking sequences if an insertion call. The resulting 'query' sequences are then mapped using minimap2 to both haplotypes. Each aligned query gives rise to a map of aligned bases $P=\{(q_1,t_1), \dots, (q_n,t_n)\}$. To score each variant call, we find two indexes i, j. These index the end of the prefix, and beginning of the suffix in the query. When the call is valid, (P [j][0] - P [i][0]) - (P [j][1] - P [j][0]) is equal to 0. To account for differences in alignment, we iteratively search for an (i^{opt}, j^{opt}) combination, with $i^{opt} \leq i$ and $j \leq j^{opt}$ that gives the smallest difference. Variants are reported as valid if the difference is less than 10 bases or the intervals defined between P [i^{opt}] and P [j^{opt}] are within 95% length in either haplotype. A summary report is then produced using an R script.

Use Cases

nibSV:

We benchmarked nibSV over the GIAB HG002 SV call set.[65] In summary, this call set was created using multiple long and short read technologies and underwent manual validation across multiple groups to ensure an overall high quality and accuracy. Using an Illumina data set from (2x250 GIAB HG002) we benchmarked true positives (i.e. SV that should be present), false positives (i.e. parental only SV that should not be present in the proband HG002), and false negatives (i.e. SV that should be present in HG002 but were not found). Using only chr 22 from HG002, nibSV with a kmer size of 23 takes around 2-4 minutes on a single thread with a 80gb bam file and the provided VCF file. We assessed our recall at different k-mer sizes which increases with the kmer size. For example, k=21 (2min 3sec) achieves 0.59 recall with a precision of 0.83. Interestingly, for insertions the recall rate increases to 0.86 with a precision of 0.86.

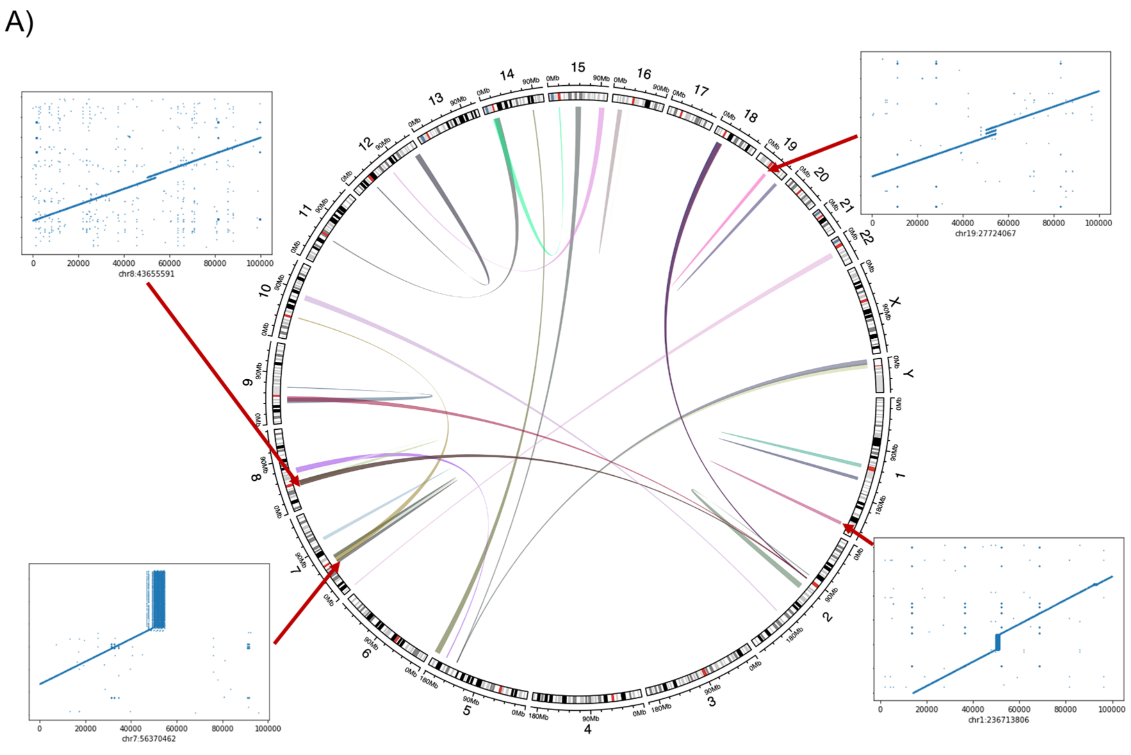
CNV2SV:

Figure 11a shows the best links (based on the alignment identity score) between CNVs identified by CNVnator and duplication SVs inferred from the dipcall alignment of CHM13 and GRCh38. The genomic areas surrounding four selected adjacent duplication events are further highlighted using dot plots, revealing the architecture of the corresponding variation in the context of the genome-genome alignment. In Figure 11b, all CNV-SV links meeting a default alignment identity threshold are shown, further revealing events corresponding to putative copy number increases of greater than two. We further explored the reason why some CNV and SV could not be linked, through statistical analysis of the raw SV calls as shown in Figure 12.

CoronaSV: We processed more than 200 SARS-CoV-2 SRA runs with CoronaSV, and Figure 13 shows the high confidence SVs generated with SURVIVOR[56] by taking a majority vote across multiple SV callers. We also looked for shared SVs across multiple samples. There were only a few inversions identified that were consistently and reliably called between samples. The inversions are likely related to the transcriptional landscape of SARS-CoV-2.[66] These inversions are small (less than 1Kb), and five of them were found in ORF1ab and one on ORF M.

CleanSV: We investigated filtering SV calls using both the curated reference set and a high-quality panel of normals (PON). The PON was created using GRIDSS calls from 3,782 normal samples freshly-sequenced at a median depth of 38x by the Hartwig Medical Foundation.[50,51] Using this PON consisting of GRIDSS calls from 3,792 freshly-sequenced normal WGS samples, we explored how the percentage of calls from short-read SV callers which were incorrectly labeled as true somatic calls, either due to being algorithmic artifacts or germline calls. Our results show the promise of such an approach. (Figure 14). Note that this is not only a check for false positives: we know a priori that many calls from somatic SV callers are mislabeled as somatic when they're actually germline. This is a known algorithmic error: SVs are normally

A)



B)

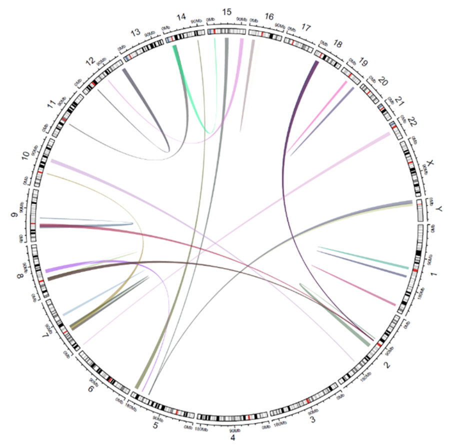


Figure 11. Relationship between the CNV calls and identified SVs across the CHM13 genome. CNV2SV results using CHM13 in comparison to GRCh38. (A) The diagram connects the genomic location of individual CNV calls on GRCh38 (broad ends) to the location of their linked SV identified in the GRCh38-CHM13 genome-genome alignment (thin ends). The diagram reveals a number of underlying putative translocation events identified for the called CNVs. Adjacent CNV-SV links (putative tandem duplications) are shown as streaks at the respective genomic position. For four select CNV-SV links, the genome-genome alignment for the underlying SV is further shown as a dot plot on the outside of the diagram. Only the best matching SV link (thin end) identified for each CNV call (broad end), as determined by sequence alignment score between both events, is shown here. (B) Similar to A, but displaying all potential CNV-SV links that meet the default alignment identity threshold of 0.8. CNVs with multiple matching duplication SV events identified in the genome-genome alignment can be explained by copy numbers greater than two occuring in distant locations, and alternatively may suggest the involvement of complex genomic rearrangements including transposable elements.

first called in the normal sample and labeled as "germline", and then the resulting SVs called using the tumor sample (separately or jointly with against the normal) are labeled as "somatic". While such an approach is common for short-read SV callers, this frequently leads to mislabeled results, often for the simple reason that the normal sample is sequenced at a much lower coverage than the tumor sample.

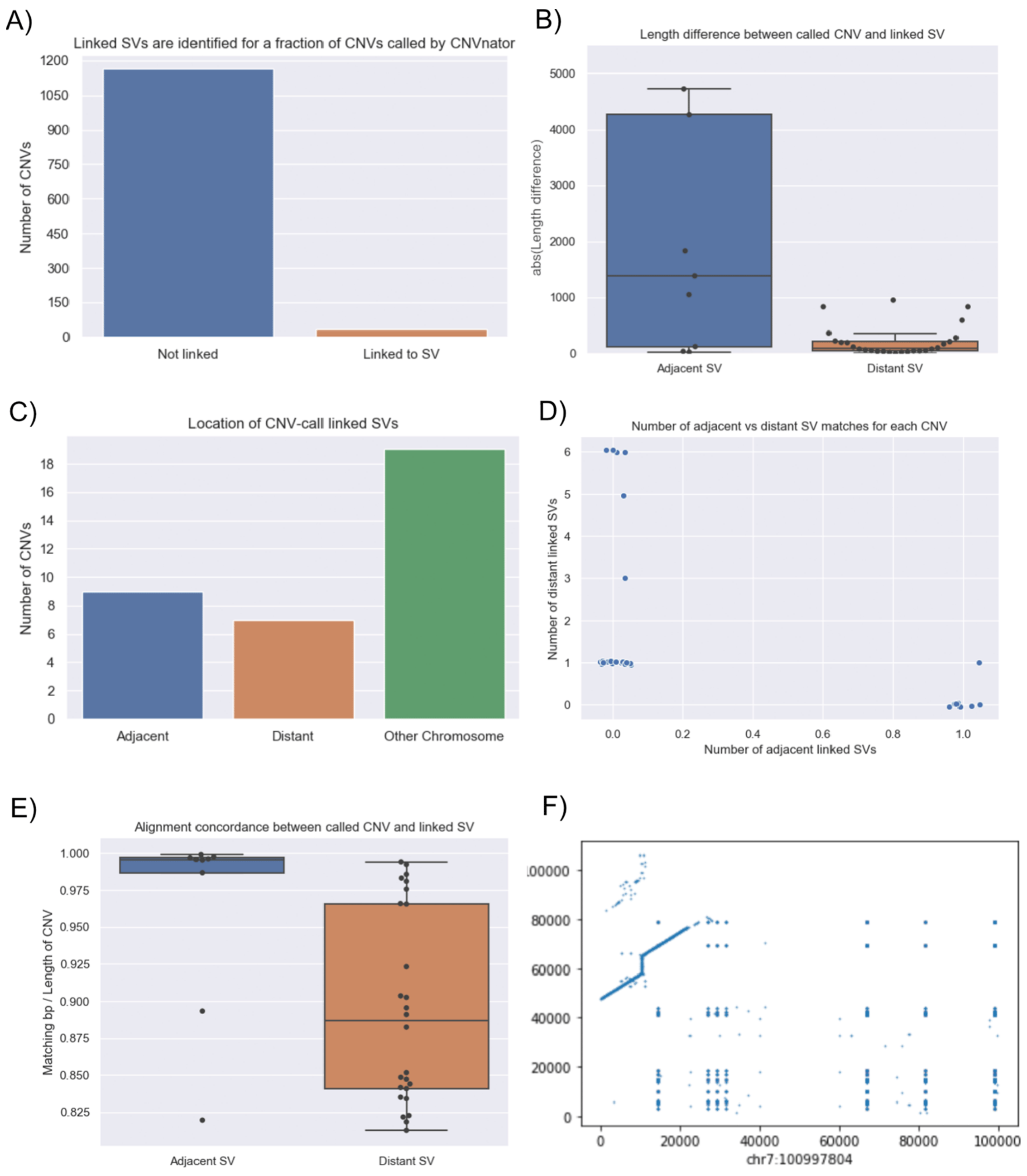


Figure 12. Summary the linkage statistics showing the characteristics of disparities between the CNV calls and SVs. In terms of linkage statistics, the majority of the CNVs identified have not been linked to a SV event, as indicated by **A**. One of the main reasons for the unsuccessful linking is the length disparity between the called CNV events and SV events as shown in **B**. Overall, among linked CNV and SV events, the distribution of three major categories are shown in **C**: adjacent events, distant events, and events spanning multiple chromosomes. Distant events are called in the case when the linked SV is at least 1Kbp away from the CNV call (either upstream or downstream). Details of the distribution of the adjacent and distant events per CNV call are given in **D** and **E** shows that alignment quality is better for adjacent matches when compared to more distant SV matches. While most links for a single CNV event are unanimously distant or adjacent, we observed an event in which a CNV was linked to both an adjacent and a distant SV which occurs on chromosome 7 (**F**: adjacent: chr7:100997804 length 8325 and distant: chr7:100994092 length 3249).

In order to release these filters/thresholds to the larger community for general usage, these filters need to be tailored specifically for specific assays. The optimal approach would be to focus on a particular use of SV calling (e.g. therapeutic oncology) with a particular assay of a certain average coverage and tumor purity. We suspect generating filters per SV

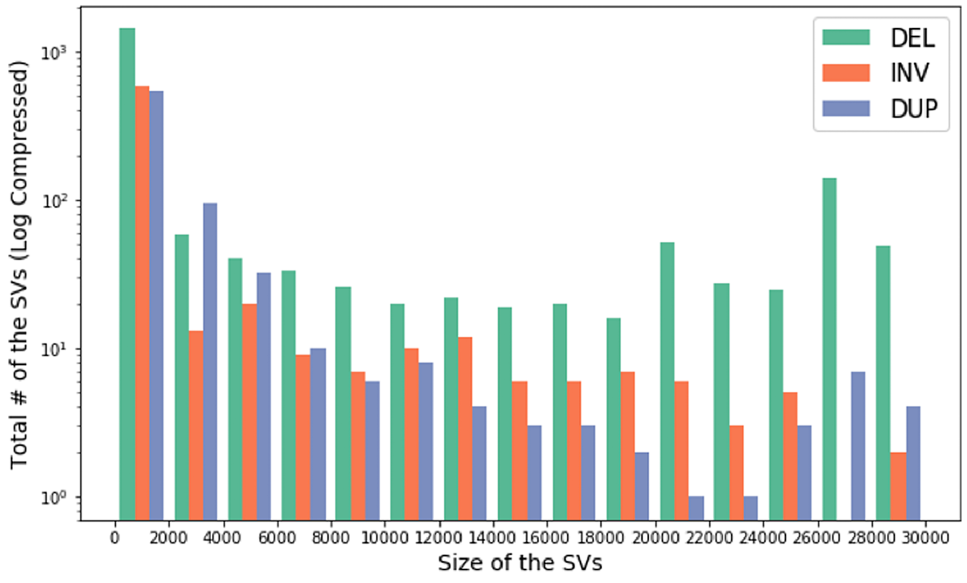


Figure 13. An analysis of size and number of identified SARS-CoV-2 SVs. Histogram showing size of the SVs and the total number of SVs across multiple Nanopore and Illumina datasets. The y-axis of the histogram is log compressed.

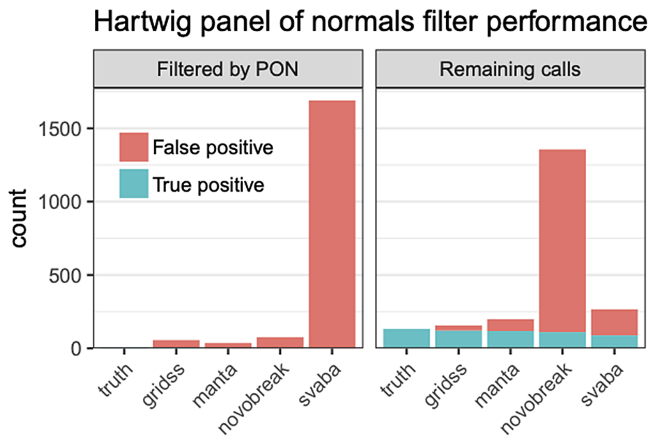


Figure 14. Filtering somatic SV calls using a Panel of Normals. Using a cohort of 3,792 freshly-sequenced WGS normal samples to create a Panel of Normals (PON) from GRIDSS calls and a set of curated calls from sample COLO829 to classify false positives, we discover that a sizeable number of false positives were found within the panel of normals, suggesting that these were miscategorized due to algorithmic errors. PON filtering based on GRIDSS calls is effective for the modern callers such as GRIDSS, Manta and SvABA which (partially) rely upon localized assembly.

caller for general use would result in minimal quality control; this is especially true within cancer genomics in terms of calling somatic and germline SVs accurately.

We plan to continue to explore whether the false positives found exhibit distinct features which we could use for future filters to distribute to the community. With these insights, given that clinical genomics still overwhelmingly relies upon short-read sequencing, our goal would be to also apply filters to all variants (e.g. the case of Mendelian diseases).

Sniphles:We used Princess (https://github.com/MeHelmy/princess) to align, detect and phase SNVs and SVs from PacBio HiFi reads 32x coverage. The resulting haplotyped bam is the input for Sniphles, where pysam (https://github.com/pysam-developers/pysam) was used for alignment. For each phase block we used the mosdepth[59] to detect coverage. Later, we called SVs using Sniffles[10,42] with the adequate numbers of reads to support SV. The identified SVs per phase block were sorted and concatenated using bcftools version 1.9,[67] and both the haplotypes were merged using SURVIVOR. Sniphles is a prototype of the idea we introduced in the manuscript and so it is still under development.

Swagg: The main results from the Swagg package includes the development of the graph module and the protein graphical application (Figure 15). The graph module is able to retrieve basic statistics from a pangenome graph in GFA format from either reference-based (fasta + VCF) or a set of assemblies (fasta), followed by conducting a pairwise comparison of all paths in the graph, and outputting a matrix in TSV format with the path names and corresponding samples in the first position. After creating the pairwise matrix, the module can plot an SV pileup over any path in the graph, counting the number of other paths that contain an SV overlapping each position. In addition to utilizing the pairwise comparisons where hotspots are references to a given sample, this approach also allows for graphs derived from vcf files. The objective from the protein graph mapping application was to show if the variants introduce new amino acids or stop a stop codon. Similarly, a pangenome graph can be constructed from DNA sequences as panproteome graphs can be built from different amino acid sequences of a protein. These graphs can then help visualize the variants between the sequences and show the paths each sequence take through the graph. Another layer of information can be added to the nodes, e.g. does a node represent a conserved or a non-conserved side, does a path divergence in the graph has any significant phenotypic characteristics, relating genome-wide association studies to these proteins graph.[68] Therefore, when aligning a new sequence to the graph, one can check the path the new sequence took in the graph and what information are related to this path.[69] Figure 15 shows an example of an annotated graph of the Nucleocapsid Phosphoprotein in SARS-COV-2.

PanOriginSV: It became apparent that the quality and representation of the clusters was a main factor in prediction accuracy. To this end, we tested PanOriginSV on a range of clusters of at least 500 sequences and only considered cases where the test sequence had a training representative in the assigned cluster. The CPU time used by PanOriginSV was 10-50x less than the linear model constructed by Plasmidhawk (depending on the cluster). We observed a range of results, with the linear prediction model outperforming PanOriginSV by up to 5% in some clusters and PanOriginSV outperforming the linear model by up to 6% in others (Table 1). With deeper analysis of the input data, we hope to achieve

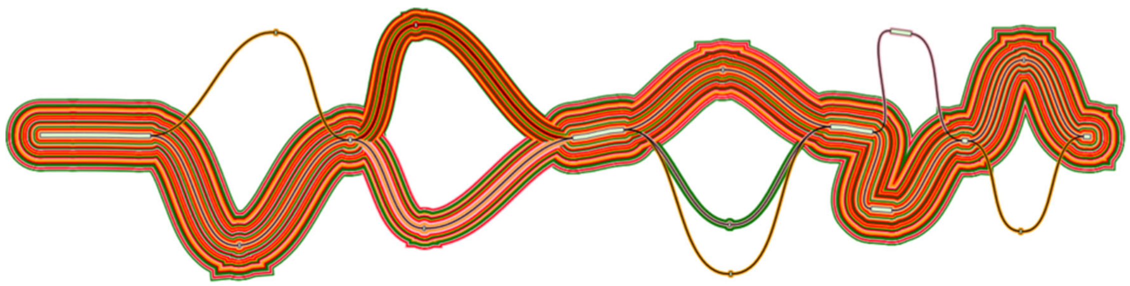


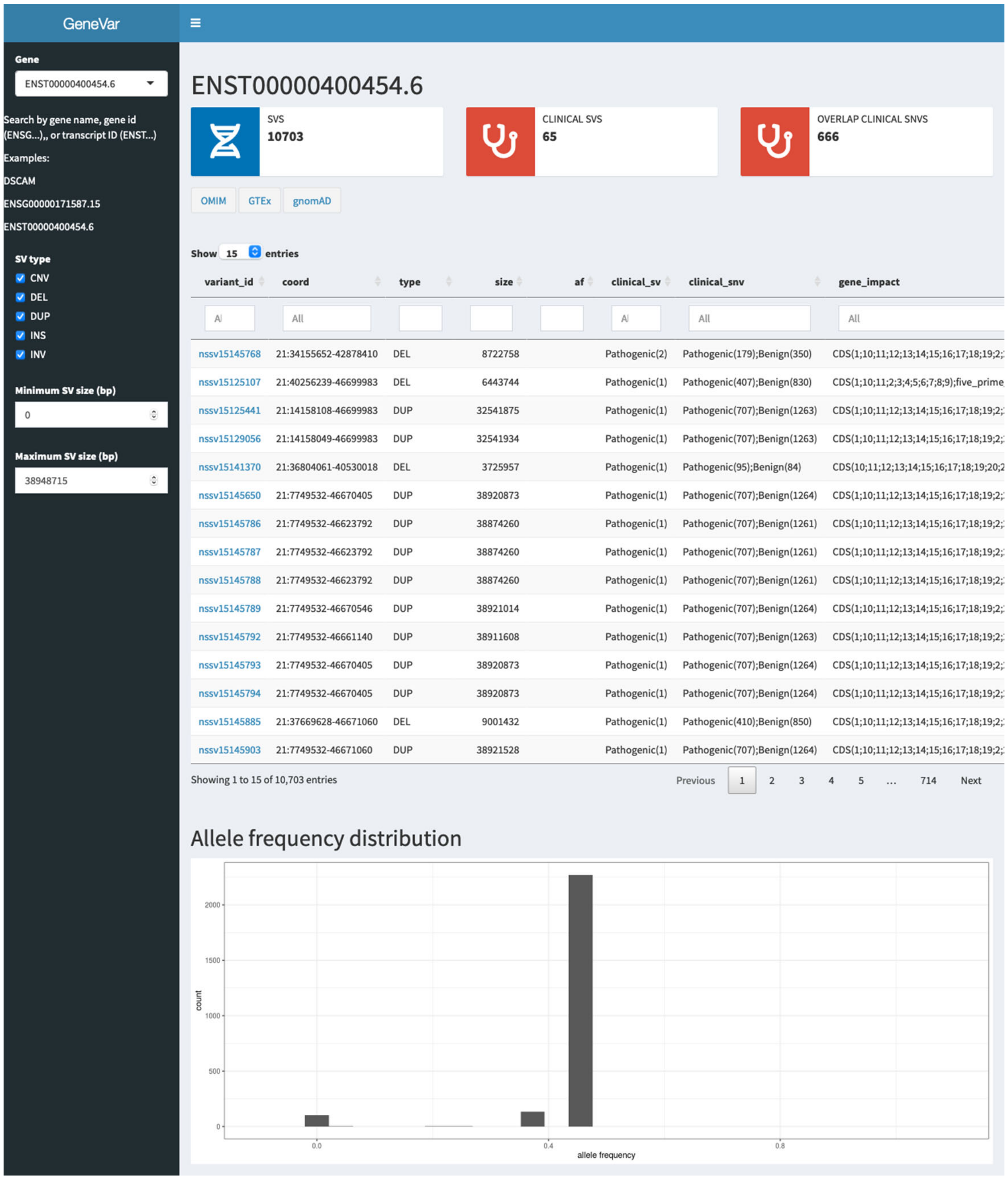
Figure 15. SARS-COV-2 nucleocapsid protein graph. Protein graph with paths representing the original sample. This graph here is a directed acyclic graph of an MSA of 26 "N" gene (Nucleocapsid Phosphoprotein) generated from 26 SARS-COV-2. Visualized using gfaviz.[70]

Table 1. Summary of PanOriginSV prediction accuracy. Benchmarking results on large clusters obtained from MMseqs2. For a single cluster, 25% of sequences were held out and used as the testing set. The accuracy of the top predicted lab was consistently higher in PanOriginSV compared to PlasmidHawk, however the accuracy when testing against the top 5 predictions for both tools was comparable.

Cluster	Train Sequences	Test Sequences	Test accuracy (Linear)	Top 5 test accuracy (Linear)	Test accuracy (Graph)	Top 5 test accuracy (Graph)
J7OEM	2870	714	0.96	0.99	0.97	0.99
3PTDM	2397	571	0.82	0.93	0.81	0.93
O3GQU	1046	275	0.65	0.88	0.71	0.83
48073	973	205	0.71	0.89	0.71	0.87
WA905	639	149	0.87	0.94	0.87	0.97
GIGX0	604	119	0.71	0.87	0.79	0.93

better clusters and thus better prediction results with the graph model. It is also worth noting that our graph construction method can be improved using more recent pangenome graph construction tools.

GeneVar: **Databrowser.** Upon querying a gene or transcript, the data browser will visualize a rare-variant burden test, allele frequency distribution, and variant level information for known SVs within dbVar. The data browser does not have a login requirement and integrates multiple public resources (Figure 16). To illustrate whether a particular gene/transcript or exon has been adequately covered to detect variation, the average depth of coverage is graphically represented. An additional panel shows gene expression levels across all general tissues included in GTEx.[71] **Report summary**. Analysis results are enriched with information from several widely used databases such as ClinVar[72] and gnomAD,[73] as well as



Gene

ENST00000400454.6

Search by gene name, gene id (ENSG...),, or transcript ID (ENST...)

Examples:

DSCAM

ENSG00000171587.15

ENST00000400454.6

SV type

✓ CNV
✓ DEL
✓ DUP
✓ INS
✓ INV

Minimum SV size (bp)

0

Maximum SV size (bp)

38948715

ENST00000400454.6

SVS 10703

CLINICAL SVS 65

OVERLAP CLINICAL SNVS 666

OMIM GTEx gnomAD

Show 15 entries

variant_id	coord	type	size	af	clinical_sv	clinical_snv	gene_impact
Al	All				Al	All	All
nssv15145768	21:34155652-42878410	DEL	8722758		Pathogenic(2)	Pathogenic(179);Benign(350)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15125107	21:40256239-46699983	DEL	6443744		Pathogenic(1)	Pathogenic(407);Benign(830)	CDS(1;10;11;2;3;4;5;6;7;8;9);five_prime
nssv15125441	21:14158108-46699983	DUP	32541875		Pathogenic(1)	Pathogenic(707);Benign(1263)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15129056	21:14158049-46699983	DUP	32541934		Pathogenic(1)	Pathogenic(707);Benign(1263)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15141370	21:36804061-40530018	DEL	3725957		Pathogenic(1)	Pathogenic(95);Benign(84)	CDS(10;11;12;13;14;15;16;17;18;19;20;2
nssv15145650	21:7749532-46670405	DUP	38920873		Pathogenic(1)	Pathogenic(707);Benign(1264)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145786	21:7749532-46623792	DUP	38874260		Pathogenic(1)	Pathogenic(707);Benign(1261)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145787	21:7749532-46623792	DUP	38874260		Pathogenic(1)	Pathogenic(707);Benign(1261)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145788	21:7749532-46623792	DUP	38874260		Pathogenic(1)	Pathogenic(707);Benign(1261)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145789	21:7749532-46670546	DUP	38921014		Pathogenic(1)	Pathogenic(707);Benign(1264)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145792	21:7749532-46661140	DUP	38911608		Pathogenic(1)	Pathogenic(707);Benign(1263)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145793	21:7749532-46670405	DUP	38920873		Pathogenic(1)	Pathogenic(707);Benign(1264)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145794	21:7749532-46670405	DUP	38920873		Pathogenic(1)	Pathogenic(707);Benign(1264)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145885	21:37669628-46671060	DEL	9001432		Pathogenic(1)	Pathogenic(410);Benign(850)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;
nssv15145903	21:7749532-46671060	DUP	38921528		Pathogenic(1)	Pathogenic(707);Benign(1264)	CDS(1;10;11;12;13;14;15;16;17;18;19;2;

Showing 1 to 15 of 10,703 entries

Previous 1 2 3 4 5 … 714 Next

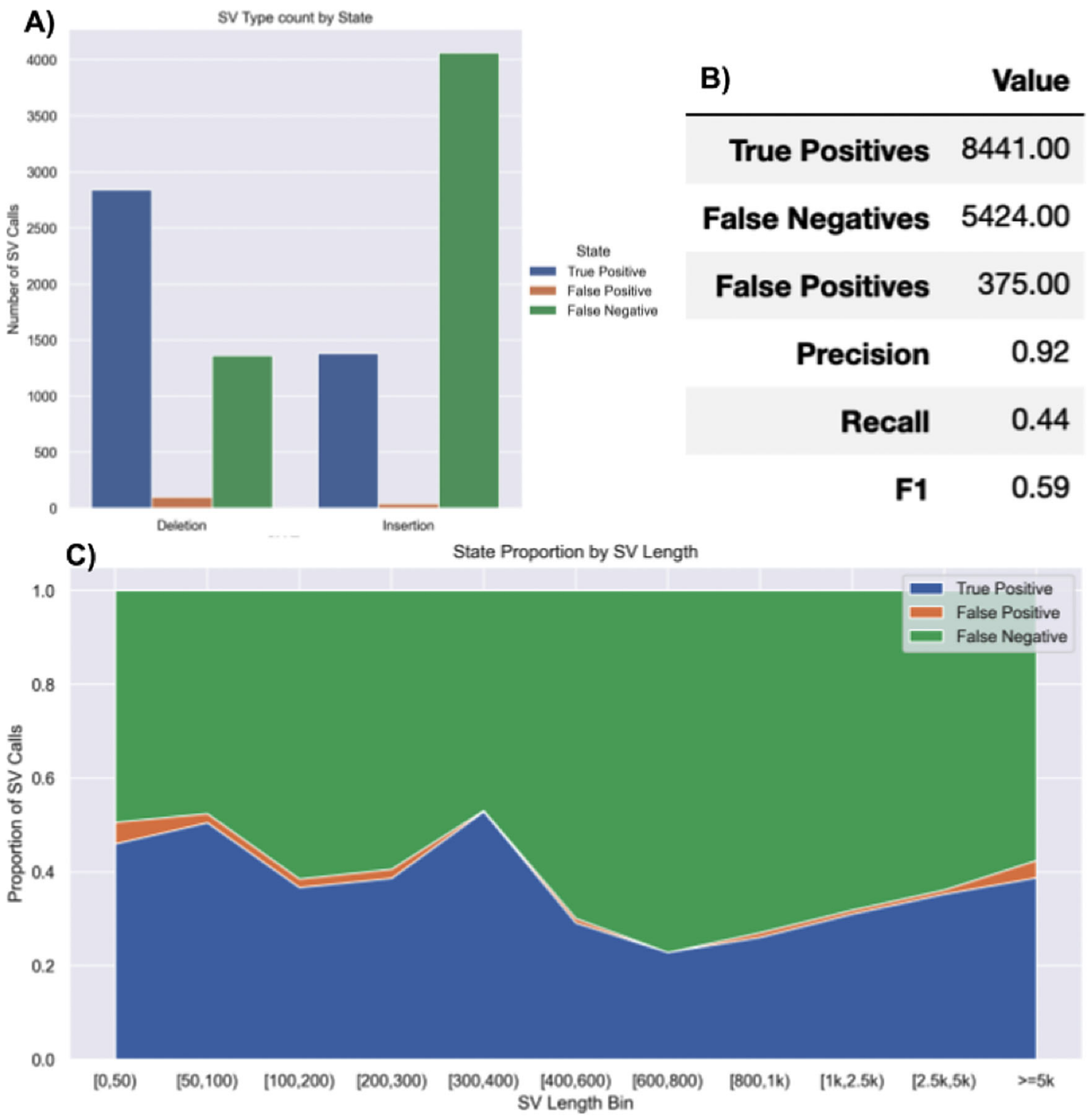
Allele frequency distribution

Figure 16. User interface of GeneVar data browser. A description of the elementary transcript details for the gene of interest. This includes the Ensembl transcript ID, Ensembl Gene ID, number of exons and genomic coordinates as described in the GRCh38 build.

graphical visualization utilities integrated in the pipeline as part of GeneVar. Resulting variants are reported in a tab delimited format to favor practical use of worksheet software such as iWork Number, Microsoft Excel or Google Spreadsheets. All information can be downloaded in tabular form.

SVTeaser: SVTeaser was able to simulate SV data on average 14 min per sample when tested on chromosome two. SVTeaser output includes organized VCF results of true positive, false positive, and false negative from evaluated SV callers. Furthermore, performance scores are reported alongside automated plots for quick visual evaluation of SV callers (Figure 17). SVTeaser is able to simulate sequencing for known deletions and insertions with various coverage, read length, and insert length. This enabled thorough evaluation for understanding the strengths and boundaries of the SV callers in question (Figure 18).

XSVLen: A diploid assembly of HG002 from Wenger *et al*[74] was used to benchmark variants. The assembly has an N50 of 16.1 and 18.0 Mb per haplotype. Variants were called using 50-fold coverage of ONT reads using CuteSV version



B)	Value
True Positives	8441.00
False Negatives	5424.00
False Positives	375.00
Precision	0.92
Recall	0.44
F1	0.59

Figure 17. Summary report output of SVTeaser. A report generated from benchmarking a real HG002 Manta 30x Illumina sample against GIAB Tier1 SVs. A) Counts of SVs by SVType and their intersection state with GIAB Tier1 benchmark SVs. B) Summary table of benchmarking performance. C) Proportions of SV intersection states with the benchmark by SV size bin.

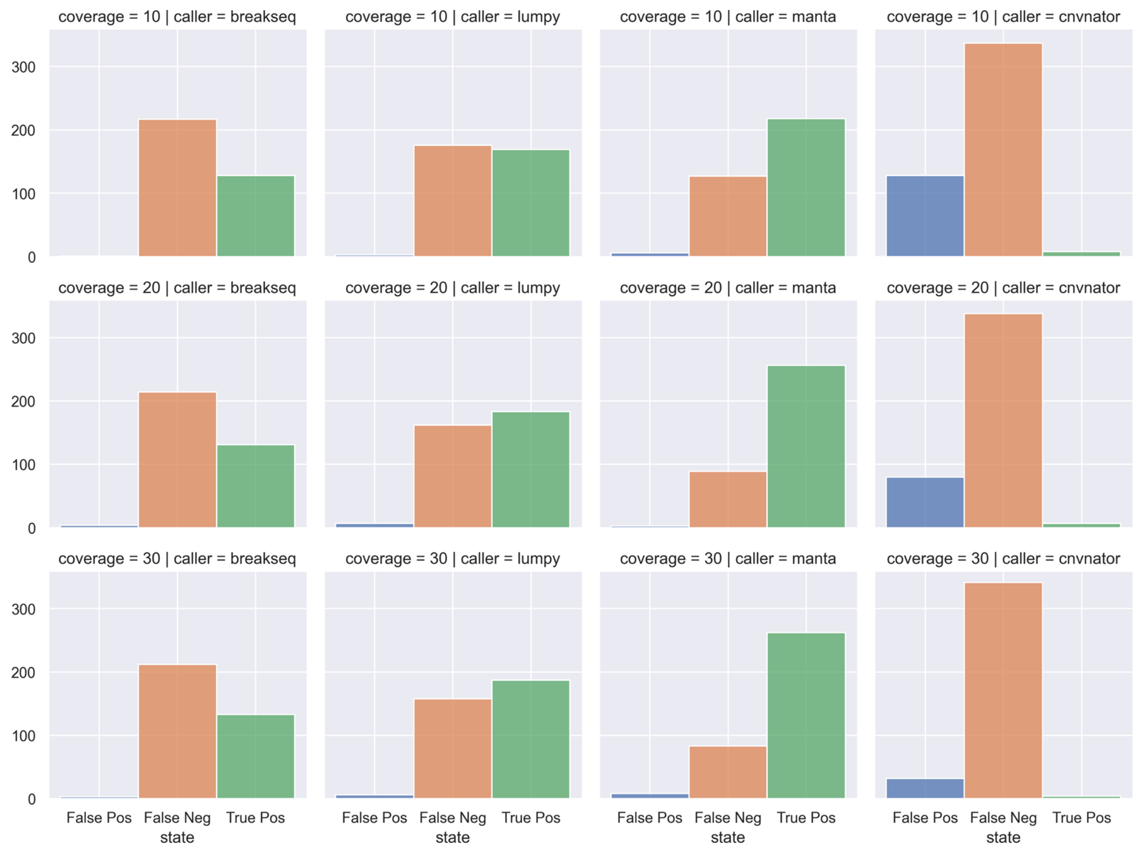


Figure 18. A comparison of simulated DELs across SV callers and sequencing coverages. Count of False Positives, False Negatives, and True Positives from four SV callers (columns) against chromosome 2 deletions through multiple simulated coverages (rows). SV callers: breakseq, lumpy, manta, cnvnator. Coverages: 10x, 20x, 30x.

Table 2. Summary: Insertion and deletion events are compared according to structural variant size and number and overlaps with the haplotyped assembly used in their performed evaluation. Truvari classification for the overlapping calls are shown as well as TP and FP found in the haplotyped assembly.

SVType	**Insertions**		**Deletions**	
SVLen	**<50**	**>=50**	**<50**	**>=50**
Number of insertion/deletion calls by cuteSV	20007	14942	38331	10926
Number of these calls that overlap assembly contigs	16054	12075	30494	4074
Number of truvari TP/FP/no-call (NP)	3351 FP 7200 TP 15974 NP	368 FP 4348 TP 7359 NP	1842 FP 18 TP 28634 NP	117 FP 1137 TP 2820 NP
Number of assembly calls TP/FP/no-call (NP)	5220 FP 10834 TP	8979 FP 8502 TP	23202 FP 7292 TP	3444 FP 3031 TP

v1.0.8. The number of INS and DEL per SV size can be observed in Table 2. The number of INS/DEL overlapping with the haplotype-resolved assemblies are shown in Table 2 as well as a comparison of assembly-based calls and gold-standard Truvari calls.

Conclusion

The results of the 2020 Baylor College of Medicine/DNANexus hackathon described here represent novel work that pushes the field forward for human genome SV detection but also for Covid related research. Both are needed to further current findings about diversity and the complexity of organisms and their genotypes. To further facilitate this progress in a FAIR-compliant manner, 80 people came together from across the world in October 2020 completed 10 groundbreaking prototypes. Hackathons like these not only represent short bursts of prototype development, but are essential to form groups and communities, inspire communication across countries and research institutions, and form novel collaboration networks. As such, this year's hackathon not only sparked the projects described here, but also highlighted the need for unified databases for SVs and other genomic features hosted on DNAnexus, Anvil, and other platforms as well as larger standards and references (e.g. GIAB, UKBB). This is essential to ensure quality standards for benchmarking and comparability, which will further advance the science and medical research.

Rapidly switching to a completely remote hackathon that enabled increased international participation was made necessary by the COVID-19 pandemic. This allowed for an open science effort across an even more diverse population of individuals and professional backgrounds. That diversity made it possible to complete 10 projects, which spearheaded novel insights in the understanding of structural variants in humans, as well as COVID19 genome structure. More importantly, it led to new synergies among participants, an active online community, and new friendships across borders.

Data availability

Associated code is available at: DOI 10.17605/OSF.IO/ME62X

Data sources utilized:

NibbleSV: Genome in a Bottle, HG002, SV callset (ftp://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/HG002_NA24385_son/PacBio_CCS_15kb/).

CNV2SV: CHM13 (https://github.com/nanopore-wgs-consortium/CHM13#telomere-to-telomere-consortium) and GRCh38.

CoronaSV: Data sources available on https://github.com/collaborativebioinformatics/coronasv.

CleanSV: hs37d5 HG002 BAM (ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/phase2_reference_assembly_sequence/hs37d5.fa.gz), hg19 COLO829 BAM (https://nextcloud.hartwigmedicalfoundation.nl/s/LTiKTd8XxBqwaiC?path=%2FHMFTools-Resources%2FGRIDSS-Purple-Linx-Docker).

SWAGG: Data sources available on https://github.com/collaborativebioinformatics/swagg/blob/main/sample_manifest.tsv.

XSVLen: HG002 haplotyped-resolved assemblies (NCBI assembly with accessions GCA_0047796458.1 (maternal) and GCA_004796285.1 (paternal)), GIAB HG002 truthset (https://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/analysis/NIST_SVs_Integration_v0.6/HG002_SVs_Tier1_v0.6.vcf.gz).

GeneVar: Data sources available on https://github.com/collaborativebioinformatics/GeneVar.

PanOriginSV: genetic engineering attribution challenge (GEAC).

https://www.drivendata.org/competitions/63/genetic-engineering-attribution/.

SVTeaser: All data utilized was based on simulations

Sniphles: Genome in a Bottle, HG002, SV callset (ftp://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/HG002_NA24385_son/PacBio_CCS_15kb/).

Software availability

NibbleSV

Source code available from: https://github.com/collaborativebioinformatics/nibSV

Archived source code at time of publication:

License: MIT license

CNVSV

Source code available from: https://github.com/collaborativebioinformatics/CNV2SV

Archived source code at time of publication:

License: MIT license

CoronaSV

Source code available from: https://github.com/collaborativebioinformatics/coronasv

Archived source code at time of publication:

License: MIT license

CleanSV

Source code available from: https://github.com/collaborativebioinformatics/CleanSV

Archived source code at time of publication:

License: MIT license

Sniphles

Source code available from: https://github.com/collaborativebioinformatics/Sniphles

Archived source code at time of publication:

License: MIT license

Swagg

Source code available from: https://github.com/collaborativebioinformatics/swagg

Archived source code at time of publication:

License: MIT license

PanOriginSV

Source code available from: https://github.com/collaborativebioinformatics/PanOriginSV

Archived source code at time of publication:

License: MIT license

GeneVar

Source code available from: https://github.com/collaborativebioinformatics/GeneVar

Archived source code at time of publication:

License: MIT license

SVTeaser

Source code available from: https://github.com/collaborativebioinformatics/SVTeaser

Archived source code at time of publication:

License: MIT license

XVSLen

Source code available from: https://github.com/collaborativebioinformatics/The_X_team

Archived source code at time of publication:

License: MIT license

Acknowledgements

The authors like to thank Richard Gibbs, Aaron Wenger and Stephen Rudd for their helpful discussions during the hackathon. In addition, Gerald Wright and Brenton Pyle for helping with advertisement and organising. DNA nexus provided computational resources for the hackathon. The authors would also like to thank Oxford Nanopore Technologies and Pacific Biosciences for sponsored prizes.

References

1. Ho SS, Urban AE, Mills RE: **Structural variation in the sequencing era.** *Nat Rev Genet.* 2020 Mar; **21**(3): 171–89.
PubMed Abstract | Publisher Full Text | Free Full Text
2. Feuk L, Carson AR, Scherer SW: **Structural variation in the human genome.** *Nat Rev Genet.* 2006 Feb; **7**(2): 85–97.
PubMed Abstract | Publisher Full Text
3. Alkan C, Coe BP, Eichler EE: **Genome structural variation discovery and genotyping.** *Nat Rev Genet.* 2011 May; **12**(5): 363–76.
PubMed Abstract | Publisher Full Text | Free Full Text
4. Sanchis-Juan A, Stephens J, French CE, *et al.*: **Complex structural variants in Mendelian disorders: identification and breakpoint resolution using short- and long-read genome sequencing.** *Genome Med.* 2018 Dec 7; **10**(1): 95.
PubMed Abstract | Publisher Full Text | Free Full Text
5. Li Y, Roberts ND, Wala JA, *et al.*: **Patterns of somatic structural variation in human cancer genomes.** *Nature.* 2020 Feb; **578**(7793): 112–21.
PubMed Abstract | Publisher Full Text | Free Full Text
6. Seaby EG, Ennis S: **Challenges in the diagnosis and discovery of rare genetic disorders using contemporary sequencing technologies.** *Brief Funct Genomics.* 2020 Jul 29; **19**(4): 243–58.
PubMed Abstract | Publisher Full Text
7. Jenko Bizjan B, Katsila T, Tesovnik T, *et al.*: **Challenges in identifying large germline structural variants for clinical use by long read sequencing.** *Comput Struct Biotechnol J.* 2020; **18**: 83–92.
PubMed Abstract | Publisher Full Text | Free Full Text
8. Wenger AM, Peluso P, Rowell WJ, *et al.*: **Accurate circular consensus long-read sequencing improves variant detection and assembly of a human genome.** *Nat Biotechnol.* 2019 Oct; **37**(10): 1155–62.
PubMed Abstract | Publisher Full Text | Free Full Text
9. Norris AL, Workman RE, Fan Y, *et al.*: **Nanopore sequencing detects structural variants in cancer.** *Cancer Biol Ther.* 2016 Jan 19; **17**(3): 246–53.
PubMed Abstract | Publisher Full Text | Free Full Text
10. Sedlazeck FJ, Rescheneder P, Smolka M, *et al.*: **Accurate detection of complex structural variations using single-molecule sequencing.** *Nat Methods.* 2018 Jun; **15**(6): 461–8.
PubMed Abstract | Publisher Full Text | Free Full Text
11. Tsang H-F, Xue VW, Koh S-P, *et al.*: **NanoString, a novel digital color-coded barcode technology: current and future applications in molecular diagnostics.** *Expert Rev Mol Diagn.* 2017 Jan; **17**(1): 95–103.
PubMed Abstract | Publisher Full Text
12. Chen S, Krusche P, Dolzhenko E, *et al.*: **Paragraph: a graph-based structural variant genotyper for short-read sequence data.** *Genome Biol.* 2019 Dec 19; **20**(1): 291.
PubMed Abstract | Publisher Full Text | Free Full Text
13. Hickey G, Heller D, Monlong J, *et al.*: **Genotyping structural variants in pangenome graphs using the vg toolkit.** *Genome Biol.* 2020 Feb 12; **21**(1): 35.
PubMed Abstract | Publisher Full Text | Free Full Text
14. Christgen M, van Luttikhuizen JL, Raap M, *et al.*: **Precise ERBB2 copy number assessment in breast cancerby means of molecular inversion probe array analysis.** *Oncotarget.* 2016 Dec 13; **7**(50): 82733–82740.
PubMed Abstract | Publisher Full Text | Free Full Text
15. Boujemaa M, Hamdi Y, Mejri N, *et al.*: **Germline copy number variationsin BRCA1/2 negative families: Role in the molecular etiology of hereditarybreast cancer in Tunisia.** *PLoS One.* 2021 Jan 27; **16**(1): e0245362.
PubMed Abstract | Publisher Full Text | Free Full Text
16. Eisenstein M: **Closing in on a complete humangenome.** *Nature.* 2021 Feb; **590**(7847): 679–681.
PubMed Abstract | Publisher Full Text
17. Islam MR, Hoque MN, Rahman MS, *et al.*: **Genome-wide analysis of SARS-CoV-2 virus strains circulating worldwide implicates heterogeneity.** *Sci Rep.* 2020 Aug 19; **10**(1): 14004.
PubMed Abstract | Publisher Full Text | Free Full Text
18. Young BE, Fong S-W, Chan Y-H, *et al.*: **Effects of a major deletion in the SARS-CoV-2 genome on the severity of infection and the inflammatory response: an observational cohort study.** *Lancet.* 2020 Aug 29; **396**(10251): 603–11.
PubMed Abstract | Publisher Full Text | Free Full Text
19. Cameron DL, Di Stefano L, Papenfuss AT: **Comprehensive evaluation and characterisation of short read general-purpose structural variant calling software.** *Nat Commun.* 2019 Jul 19; **10**(1): 3240.
PubMed Abstract | Publisher Full Text | Free Full Text
20. McCrone JT, Lauring AS: **Measurements of Intrahost Viral Diversity Are Extremely Sensitive to Systematic Errors in Variant Calling.** *J Virol.* 2016 Aug 1; **90**(15): 6884–95.
PubMed Abstract | Publisher Full Text | Free Full Text

21. Weißbach S, Sys S, Hewel C, *et al.*: **Reliability of genomic variants across different next-generation sequencing platforms and bioinformatic processing pipelines.** *BMC Genomics.* 2021 Jan 19; **22**(1): 62.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

22. Sandmann S, Karimi M, de Graaf AO, *et al.*: **appreci8: a pipeline for precise variant calling integrating 8 tools.** *Bioinformatics.* 2018 Dec 15; **34**(24): 4205–12.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

23. Bamshad MJ, Ng SB, Bigham AW, *et al.*: **Exome sequencing as a tool for Mendelian disease gene discovery.** *Nat Rev Genet.* 2011 Sep 27; **12**(11): 745–55.
PubMed Abstract | **Publisher Full Text**

24. Liu H-Y, Zhou L, Zheng M-Y, *et al.*: **Diagnostic and clinical utility of whole genome sequencing in a cohort of undiagnosed Chinese families with rare diseases.** *Sci Rep.* 2019 Dec 18; **9**(1): 19365.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

25. Murciano-Goroff YR, Taylor BS, Hyman DM, *et al.*: **Toward a More Precise Future for Oncology.** *Cancer Cell.* 2020 Apr 13; **37**(4): 431–42.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

26. Donoghue MTA, Schram AM, Hyman DM, *et al.*: **Discovery through clinical sequencing in oncology.** *Nature Cancer.* 2020 Aug 10; **1**(8): 774–83.
Publisher Full Text

27. Mahmoud M, Gobet N, Cruz-Dávalos DI, *et al.*: **Structural variant calling: the long and the short of it.** *Genome Biol.* 2019 Nov 20; **20**(1): 246.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

28. Chaisson MJP, Sanders AD, Zhao X, *et al.*: **Multi-platform discovery of haplotype-resolved structural variation in human genomes.** *Nat Commun.* 2019 Apr 16; **10**(1): 1784.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

29. Ben-David U, Amon A: **Context is everything: aneuploidy in cancer.** *Nat Rev Genet.* 2020 Jan; **21**(1): 44–62.
PubMed Abstract | **Publisher Full Text**

30. Martin M, Patterson M, Garg S, *et al.*: **WhatsHap: fast and accurate read-based phasing.** *bioRxiv.* 2016 [cited 2018 Oct 23]. p. 085050.
Reference Source

31. Majidian S, Sedlazeck FJ: **PhaseME: Automatic rapid assessment of phasing quality and phasing improvement.** *Gigascience.* 2020 Jul1; **9**(7).
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

32. Sedlazeck FJ, Lee H, Darby CA, *et al.*: **Piercing the dark matter: bioinformatics of long-range sequencing and mapping.** *Nat Rev Genet.* 2018 Jun; **19**(6): 329–46.
PubMed Abstract | **Publisher Full Text**

33. Edge P, Bansal V: **Longshot enables accurate variant calling in diploid genomes from single-molecule long read sequencing.** *Nat Commun.* 2019 Oct 11; **10**(1): 4660.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

34. Jinek M, Chylinski K, Fonfara I, *et al.*: **A programmable dual-RNA-guided DNA endonuclease in adaptive bacterial immunity.** *Science.* 2012 Aug 17; **337**(6096): 816–21.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

35. Wang Q, Liu TR, Leo Elworth RA, *et al.*: **PlasmidHawk: Alignment-based Lab-of-Origin Prediction of Synthetic Plasmids.**Cold Spring Harbor Laboratory; 2020 [cited 2021 Jan 13]. p. 2020.05.22.110270.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

36. Chikhi R, Limasset A, Jackman S, *et al.*: **On the representation of de Bruijn graphs.** *J Comput Biol.* 2015 May; **22**(5): 336–52.
PubMed Abstract | **Publisher Full Text**

37. Rautiainen M, Mäkinen V, Marschall T: **Bit-parallel sequence-to-graph alignment.** *Bioinformatics.* 2019 Oct 1; **35**(19): 3599–607.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

38. Steinegger M, Söding J: **MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets.** *Nat Biotechnol.* 2017 Nov; **35**(11): 1026–8.
PubMed Abstract | **Publisher Full Text**

39. Ganel L, Abel HJFinMetSeq Consortium, *et al.*: **SVScore: an impact prediction tool for structural variation.** *Bioinformatics.* 2017 Apr 1; **33**(7): 1083–5.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

40. Kumar S, Harmanci A, Vytheeswaran J, *et al.*: **SVFX: a machine learning framework to quantify the pathogenicity of structural variants.** *Genome Biol.* 2020 Nov 9; **21**(1): 1–21.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

41. Zook JM, Catoe D, McDaniel J, *et al.*: **Extensive sequencing of seven human genomes to characterize benchmark reference materials.** *Sci Data.* 2016 Jun 7; **3**: 160025.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

42. Jiang T, Liu Y, Jiang Y, *et al.*: **Long-read-based human genomic structural variation detection with cuteSV.** *Genome Biol.* 2020 Aug 3; **21**(1): 1–24.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

43. Cameron DL, Schröder J, Penington JS, *et al.*: **GRIDSS: sensitive and specific genomic rearrangement detection using positional de Bruijn graph assembly.** *Genome Res.* 2017 Dec; **27**(12): 2050–60.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

44. Chen X, Schulz-Trieglaff O, Shaw R, *et al.*: **Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications.** *Bioinformatics.* 2016 Apr 15; **32**(8): 1220–2.
PubMed Abstract | **Publisher Full Text**

45. Rausch T, Zichner T, Schlattl A, *et al.*: **DELLY: structural variant discovery by integrated paired-end and split-read analysis.** *Bioinformatics.* 2012 Sep 15; **28**(18): i333–9.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

46. Wala JA, Bandopadhayay P, Greenwald NF, *et al.*: **SvABA: genome-wide detection of structural variants and indels by local assembly.** *Genome Res.* 2018 Apr; **28**(4): 581–91.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

47. Valle-Inclan JE, Besselink NJM, de Bruijn E, *et al.*: **A multi-platform reference for somatic structural variation detection.** *Cold Spring Harbor Laboratory.* 2020 [cited 2021 Feb 2]. p. 2020.10.15.340497.
Reference Source

48. Belyeu JR, Chowdhury M, Brown J, *et al.*: **Samplot: A Platform for Structural Variant Visual Validation and Automated Filtering.** *Cold Spring Harbor Laboratory.* 2020 [cited 2021 Jan 13]. p. 2020.09.23.310110.
Reference Source

49. Geoffroy V, Herenger Y, Kress A, *et al.*: **AnnotSV: an integrated tool for structural variations annotation.** *Bioinformatics.* 2018 Oct 15; **34**(20): 3572–4.
PubMed Abstract | **Publisher Full Text**

50. Priestley P, Baber J, Lolkema MP, *et al.*: **Pan-cancer whole-genome analyses of metastatic solid tumours.** *Nature.* 2019 Oct 23; **575**(7781): 210–6.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

51. Cameron DL, Baber J, Shale C, *et al.*: **GRIDSS2: harnessing the power of phasing and single breakends in somatic structural variant detection.** *Cold Spring Harbor Laboratory.* 2020 [cited 2021 Feb 2]. p. 2020.07.09.196527.
Reference Source

52. Bolger AM, Lohse M, Usadel B: **Trimmomatic: a flexible trimmer for Illumina sequence data.** *Bioinformatics.* 2014 Aug 1; **30**(15): 2114–20.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

53. Li H: **Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM.** *arXiv [q-bio.GN].* 2013.
Reference Source

54. Layer RM, Chiang C, Quinlan AR, *et al.*: **LUMPY: a probabilistic framework for structural variant discovery.** *Genome Biol.* 2014 Jun 26; **15**(6): R84.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

55. Soylev A, Kockan C, Hormozdiari F, *et al.*: **Toolkit for automated and rapid discovery of structural variants.** *Methods.* 2017 Oct 1; **129**: 3–7.
PubMed Abstract | **Publisher Full Text**

56. Jeffares DC, Jolly C, Hoti M, *et al.*: **Transient structural variations have strong effects on quantitative traits and reproductive isolation in fission yeast.** *Nat Commun.* 2017 Jan 24; **8**: 14061.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

57. Kent WJ, Sugnet CW, Furey TS, *et al.*: **The human genome browser at UCSC.** *Genome Res.* 2002 Jun; **12**(6): 996–1006.
Reference Source

58. Pedersen BS, Quinlan AR: **cyvcf2: fast, flexible variant analysis with Python.** *Bioinformatics.* 2017 Jun 15; **33**(12): 1867–9.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

59. Pedersen BS, Quinlan AR: **Mosdepth: quick coverage calculation for genomes and exomes.** *Bioinformatics.* 2018 Mar 1; **34**(5): 867–8.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

60. Hoffman EA, McCulley A, Haarer B, *et al.*: **Break-seq reveals hydroxyurea-induced chromosome fragility as a result of unscheduled conflict between DNA replication and transcription.** *Genome Res.* 2015 Mar; **25**(3): 402–12.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

61. Li H: **Tabix: fast retrieval of sequence features from generic TAB-delimited files.** *Bioinformatics.* 2011 Mar 1; **27**(5): 718–9.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

62. Lappalainen I, Lopez J, Skipper L, *et al.*: **DbVar and DGVa: public archives for genomic structural variation.** *Nucleic Acids Res.* 2013

Jan; **41**(Database issue): D936–41.
PubMed Abstract | Publisher Full Text | Free Full Text

63. Huang W, Li L, Myers JR, *et al.*: **ART: a next-generation sequencing read simulator.** *Bioinformatics.* 2012 Feb 15; **28**(4): 593–4.
PubMed Abstract | Publisher Full Text | Free Full Text

64. Zarate S, Carroll A, Mahmoud M, *et al.*: **Parliament2: Accurate structural variant calling at scale.** *Gigascience.* 2020 Dec 21; **9**(12).
PubMed Abstract | Publisher Full Text | Free Full Text

65. Zook JM, Hansen NF, Olson ND, *et al.*: **A robust benchmark for detection of germline large deletions and insertions.** *Nat Biotechnol.* 2020 Nov; **38**(11): 1347–55.
PubMed Abstract | Publisher Full Text

66. Sapoval N: **SARS-CoV-2 genomic diversity and the implications for qRT-PCR diagnostics and transmission.** *Genome Res.* 2021; **31**(4): 635–644.

67. Li H, Handsaker B, Wysoker A, *et al.*: **The Sequence Alignment/Map format and SAMtools.** *Bioinformatics.* 2009 Aug 15; **25**(16): 2078–9.
PubMed Abstract | Publisher Full Text | Free Full Text

68. Jaillard M, Lima L, Tournoud M, *et al.*: **A fast and agnostic method for bacterial genome-wide association studies: Bridging the gap between k-mers and genetic events.** *PLoS Genet.* 2018 Nov; **14**(11): e1007758.
PubMed Abstract | Publisher Full Text | Free Full Text

69. Lee C, Grasso C, Sharlow MF: **Multiple sequence alignment using partial order graphs.** *Bioinformatics.* 2002 Mar; **18**(3): 452–64.
PubMed Abstract | Publisher Full Text

70. Gonnella G, Niehus N, Kurtz S: **GfaViz: flexible and interactive visualization of GFA sequence graphs.** *Bioinformatics.* 2019 Aug 15; **35**(16): 2853–5.
PubMed Abstract | Publisher Full Text

71. GTEx Consortium: **The Genotype-Tissue Expression (GTEx) project.** *Nat Genet.* 2013 Jun; **45**(6): 580–5.
PubMed Abstract | Publisher Full Text | Free Full Text

72. Landrum MJ, Chitipiralla S, Brown GR, *et al.*: **ClinVar: improvements to accessing data.** *Nucleic Acids Res.* 2020 Jan 8; **48**(D1): D835–44.
PubMed Abstract | Publisher Full Text | Free Full Text

73. Karczewski KJ, Francioli LC, Tiao G, *et al.*: **The mutational constraint spectrum quantified from variation in 141,456 humans.** *Nature.* 2020 May; **581**(7809): 434–43.
PubMed Abstract | Publisher Full Text | Free Full Text

74. Chin C-S, Wagner J, Zeng Q, *et al.*: **A diploid assembly-based benchmark for variants in the major histocompatibility complex.** *Nat Commun.* 2020 Sep 22; **11**(1): 4794.
PubMed Abstract | Publisher Full Text | Free Full Text

Open Peer Review

Current Peer Review Status: ✓ ? ? ?

Version 2

Reviewer Report 28 September 2021

https://doi.org/10.5256/f1000research.76830.r93372

✓

Francois Sabot
DIADE, Univ Montpellier, IRD, CIRAD, Montpellier, France

The current version of the McCartney paper on the virtual hackathon is ok for me. I just saw a few formatting errors.

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Genomics, bioinformatics, evolution

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Version 1

Reviewer Report 07 June 2021

https://doi.org/10.5256/f1000research.54651.r82807

?

Loren L. Flynn
Murdoch University, Murdoch, WA, Australia

McCartney and colleagues describe a compelling and comprehensive account of the outcomes of an international hackathon to develop new tools for the discovery and analysis of structural variants (SVs). The authors describe 10 novel software tools to overcome gaps in the available tools to more accurately identify, genotype and map SVs, as well as provide predictive functional effects of the variants to gene expression. The identification of new and accurate tools to identify SVs is of critical importance for understanding the genetic underpinnings of complex disease with no monogenic pattern of disease inheritance. This manuscript is the product of a virtual international hackathon, describing newly developed tools as a foundation for future discussion and development, and the links to the already available online tools provide additional and necessary information for understanding the software applications and outputs. I recommend this manuscript for indexing following amendments. From a functional genetics perspective, I provide the following comments and suggestions:

1. The manuscript does not provide full detail about the code, methods and analysis of the individual tools, however, the purpose of the manuscript is to describe the hackathon and the resultant projects, with the intention for future development of these tools. This could be made clearer in the text with the development status and future requirements for each tool discussed.

2. A summary table of the main applications, benefits, and stage of development for each of the tools and the future direction would be helpful for readers. In particular, more information on the specific applications of each of the tools would be helpful for those reading the manuscript without a bioinformatics background.

3. The manuscript is written with an introduction to each tool, the operation for each and finishing with use cases or results for each. As a suggestion, describing each tool separately under its own heading and incorporating each of these sections would improve the flow of the manuscript.

4. The current tools for accurate SV calling and genotyping from GWAS data are limited for poly-nucleotide variations. Do any of the described tools address this limitation?

5. The Swagg software describes a tool for SV calling and mapping at both the gene and protein level. However, the advantage of SV calling and mapping at the protein level is not clear given that many SVs are found in non-coding regions. The protein function is not further described in the methods and the interpretation of the figure in the results section was not clear. It would be useful to provide more information for the protein mapping. Can this software show predictive changes to mRNA or protein structures induced by SVs in non-coding regions?

6. The GenVar tool offers a clinically relevant analysis with predictions to SV effects on regulation of gene expression. Does the tool identify SV regions based only on the reference sequence or is it able to further compare a patient sequence to the RefSeq and provide information as to the carriage of potential risk variants/variants known to effect gene expression?

7. There is no figure displaying the result output for the phase sequencing analysis Sniphles tool, this would help in explaining the data analysis and output.

Is the rationale for developing the new software tool clearly explained?

Yes

Is the description of the software tool technically sound?

Partly

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Partly

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Partly

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Genetic therapies, functional genetics

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Reviewer Report 20 May 2021

https://doi.org/10.5256/f1000research.54651.r83790

?

Ricardo Assunção Vialle
Nash Family Department of Neuroscience & Friedman Brain Institute, Icahn School of Medicine at Mount Sinai, New York, NY, USA

This manuscript describes ten Structural Variation related software tools developed at the Second Baylor College of Medicine & DNAnexus hackathon. These tools successfully cover relevant topics in the SV field and certainly will be appreciated by the scientific community. As other reviewers pointed out, my primary concerns are that the text still needs some work to improve readability and fix inconsistencies (e.g., American vs. British spelling). Also, a table summarizing all tools described will be very welcome. There are also some issues specific to some tools that should be addressed. Comments for each tool are described below.

NibbleSV: The tool seems great, performs efficiently, and the results are impressive. I just have few comments that would be interesting to see. First, the authors could add more details about the tests using the HG002. If I understood right, the GiaB tier 1 v0.6 was used as candidates to genotyping using the Illumina reads (250bp), is that correct? Also, how was the evaluation done? Second, if possible, it would be interesting to see how NibbleSV compares with other tools (e.g., paragraph). Finally, it would be nice to see how it performs in more complicated regions (e.g., segmental duplications). I wonder if NibbleSV could help filter artifact SVs found by misalignment in those regions.

CNV2SV: The motivation and the tool description must be improved in the manuscript. Also, the results are not well described and discussed, being restricted to the legends of Figures 11 and 12.

Minor:
In the methods, CNV2SV circularlize should be circlize to refer to the R package.
Missing description of letter C in figure 11 legends.

CoronaSV: The method proposes a Snakemake pipeline for SV discovery in SARS-CoV-2 genomes. It integrates short and long reads SV discovery tools, as well as assembly-based methods. CoronaSV also handles downloading sequencing data from SRA for variant calling. As a use case, the pipeline was applied to over 200 genomes, and a basic report of the results was shown. I'm not familiar with viral genomes, but I'm wondering what is the actual gain of applying such complex pipeline instead of performing *de novo* assembly and getting the SVs from it instead.

CleanSV: The tool proposes a systematic way for filtering false-positive somatic SVs in cancer genomics. The approach uses germline SVs identified in normal tissue samples to find the best filtering thresholds for specific SV callers. There is no description of how to run the tool, and an example dataset could be added to the GitHub page.

Sniphles: More details could be added to the diagram of the pipeline, the calling and merging steps are oversimplified. Is there a reason why Sniffles need Mosdepth to estimate its parameters? It would be great to see how it compares with the “first” approach of phasing mentioned by the authors, using tagged reads to phase SVs.

Swagg: The motivation and methods are not very clear in the manuscript. It seems to me that the protein-graph is basically capturing small-variants. Where SVs fit in this context? The example showed in Figure 15 needs a more detailed explanation (e.g., what the color mean? Is there any SV affecting this protein?). Also, some phrases are repeated in the introduction and methods sections.

PanOriginSV: I'm unfamiliar with the topic addressed by this tool. For the little information I could get, the proposal seems very relevant, and the results look promising. In the introduction, the authors discuss the drawbacks of using machine learning approaches and highlight PlasmidHawk advantages, but in the end, propose another machine learning method? What is different in this method compared to existing ones? I'm wondering if extending that tool to work non-linear pangenomes would be possible. Also, it would be nice to see the same benchmarking against PlasmidHawk.

GeneVar: The proposal of GeneVar seems interesting and novel. However, not all features described in the manuscript seemed to be implemented. For instance, the authors state that a rare-variant burden test is performed but, I could not find that on the demo website. Also, the annotations are still very basic, the presentation of the overlap is not user-friendly. I would suggest improving how information is shown, with more summarized information about the overlap.

SVTeaser: SVTeaser is a framework for benchmarking SVs using simulations. Simulations are performed using SURVIVOR. Overall, the method seems well developed, and the contribution is relevant for the field.

XSVLen: The methods sound truly relevant and novel. I'm very interested in applying this method in the future. I'm wondering if a more refined report could be implemented. It would be interesting to see results summarized for each haplotype separately. In the manuscript, the results table is hard to understand. For example, the first row shows 14942 INS >=50, while the second row shows a higher number when I would expect just a subset based on the description.

Is the rationale for developing the new software tool clearly explained?
Yes

Is the description of the software tool technically sound?
Partly

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Partly

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Partly

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Bioinformatics, genomics, genetics

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 14 Jun 2021

Ann Mc Cartney, NIH, Washington, USA

We would like to thank Ricardo for their time on giving the authors of this manuscript feedback and comments. Upon receiving these comments the corresponding authors reached out to the developers of each of the tools developed as part of this hackathon, each of which have done their utmost to respond to each comment or question.

Comment 1: General: This manuscript describes ten Structural Variation related software tools developed at the Second Baylor College of Medicine & DNAnexus hackathon. These tools successfully cover relevant topics in the SV field and certainly will be appreciated by the scientific community. As other reviewers pointed out, my primary concerns are that the text still needs some work to improve readability and fix inconsistencies (e.g., American vs. British spelling). Also, a table summarizing all tools described will be very welcome. There are also some issues specific to some tools that should be addressed.

Response: We agree that the format of the manuscript is somewhat cumbersome to read, however this is a pre-requisite determined by the journal for manuscripts associated with hackathons.

Comment 2: For the Developer of NibbleSV: The tool seems great, performs efficiently, and the results are impressive. I just have a few comments that would be interesting to see. First, the authors could add more details about the tests using the HG002. If I understood correctly, the GiaB tier 1 v0.6 was used as a candidate for genotyping using the Illumina reads (250bp), is that correct? Also, how was the evaluation done? Second, if possible, it would be interesting to see how NibbleSV compares with other tools (e.g., paragraph). Finally, it would be nice to see how it performs in more complicated regions (e.g., segmental duplications). I wonder if NibbleSV could help filter artifact SVs found by misalignment in those regions.

Response: Yes, the tests were done using GIAB Tier 1 v0.6 truth SV set and alignment file of illumina 2x250 bp dataset for HG002 sample. The true positive count is the number of genotyped SVs that are not parental SVs i.e. HG2count > 0 and the FP count is the parental SVs i.e. HG2count=0. The parental exclusive SVs that are not genotyped are counted as TNs and the HG002 SVs that are not genotyped by nibSV are counted as FN. The following metrics are used to evaluate performance of nibSV:

Recall = TP/ (TP+FN); Precision = TP/(TP+FP); Accuracy = (TP+TN)/(TP+FP+FN+TN); False discovery rate = FP/(TP+FP). The performance of nibSV was compared with SVTyper. SVTyper can genotype only non-insertion (ie. here deletions) type SVs, so we compared nibSV with SVtyper only for deletion type of SVs. nibSv was run with parameters k=23 and m=0.
Tool,TP,FP,FN,TN,Recall,Precision,Accuracy,FDR
nibSV,12152,2055,17957,5248,0.4,0.86,0.47,0.14
SVTyper,5873,212,24236,7091,0.20,0.97,0.35,0.03

The high recall score of nibSV shows that it outperformed SVTyper for genotyping the non-insertion type SVs. The high precision rate and low false discovery rate of SVTyper is due to the very low number of total genotyped SVs. The Paragraph tools required to add the padding bases to the SVs in the truth set. We could not successfully run Paragraph without making any changes to the truth set. Therefore, we have not included that tool in this result.

Comment 3: *For the CNV2SV Developers: The motivation and the tool description must be improved in the manuscript. Also, the results are not well described and discussed, being restricted to the legends of Figures 11 and 12. Minor:In the methods, CNV2SV circularlize should be circlize to refer to the R package.Missing description of letter C in figure 11 legends.*

Response: Thank you for your assessment. We have restructured the respective sections in the manuscript in order to clarify the tool motivation and description of operation. We have also corrected the figure accordingly.

Comment 4: For the CoronaSV Developers: The method proposes a Snakemake pipeline for SV discovery in SARS-CoV-2 genomes. It integrates short and long reads SV discovery tools, as well as assembly-based methods. CoronaSV also handles downloading sequencing data from SRA for variant calling. As a use case, the pipeline was applied to over 200 genomes, and a basic report of the results was shown. I'm not familiar with viral genomes, but I'm wondering what is the actual gain of applying such a complex pipeline instead of performing de novo assembly and getting the SVs from it instead.

Response: The reviewer makes a good point with respect to de novo assembly from short reads, which would accurately capture small structural variants. However, given SARS-CoV-2 is sequenced from an intrahost population, which often is from metatranscriptomes (RNA-sequencing of a microbiome sample) containing subgenomic RNAs, SVs reflect the transcriptional landscape of SARS-CoV-2. Thus we feel it is of use to capture the concordance across short reads, long reads, and assembly for structural variants calling.

Comment 5: For the CleanSV Developers: The tool proposes a systematic way for filtering false-positive somatic SVs in cancer genomics. The approach uses germline SVs identified in normal tissue samples to find the best filtering thresholds for specific SV callers. There is no description of how to run the tool, and an example dataset could be added to the GitHub page.

Response: Thank you to the reviewer for the response. As mentioned in the previous reviews, our contribution to the hackathon was a proof of concept. Releasing the filters for general use would require more work specifically investigating tumors of a given sequencing depth and tumor content. The panel of normals used for the paper are accessible, as cited in the paper.

Comment 6: For the Sniphles Developers: More details could be added to the diagram of the pipeline, the calling and merging steps are oversimplified. Is there a reason why Sniffles need Mosdepth to estimate its parameters? It would be great to see how it compares with the "first'' approach of phasing mentioned by the authors, using tagged reads to phase SVs**.**

Response: We would like to thank the reviewer for the comments and questions. We agree with the reviewer regarding the merging step, while this is a prototype of the idea, we will be working on enhancing the merging step. Additionally, we stated that in the manuscript.

Comment 7: For the Sniphles Developers: "Sniffles need Mosdepth to estimate its parameters''. Response: Sniffles use a minimum of 10 reads to call an SV, we suggest to use mosdopth to calculate the coverage for the region we call SVs in, so it could adapt to low and high coverage as well as technology used for sequencing (PacBio or Oxford Nanopore Technology).
Comment 8: For the Sniphles Developers: "It would be great to see how it compares with the

“first'' approach of phasing mentioned by the authors, using tagged reads to phase SVs.”.

Response: We will be working on that, as the second approach still needs more tuning.

Comment 9: For the Swagg Developers: The motivation and methods are not very clear in the manuscript. It seems to me that the protein-graph is basically capturing small-variants. Where do SVs fit in this context? The example shown in Figure 15 needs a more detailed explanation (e.g What does the color mean? Is there any SV affecting this protein?). Also, some phrases are repeated in the introduction and methods sections.

Response: We would like to thank the reviewer for their comments and questions regarding the manuscript. In response to their questions, the different colors are associated with different sequences used to build the graph, with each sequence taking a certain path in the graph, which yields the colors.

Comment 10: For the PanOriginSV Developers: I’m unfamiliar with the topic addressed by this tool. For the little information I could get, the proposal seems very relevant, and the results look promising. In the introduction, the authors discuss the drawbacks of using machine learning approaches and highlight PlasmidHawk advantages, but in the end, propose another machine learning method? What is different in this method compared to existing ones? I’m wondering if extending that tool to work non-linear pangenomes would be possible. Also, it would be nice to see the same benchmarking against PlasmidHawk.

Response: We’d first like to thank the reviewer for their feedback. While PanOriginSV does use machine learning, it has the advantage that random forests are much easier to inspect, train, and interpret than other ML models. In addition, the step of constructing the pangenome already constructs features of the sequence. The results of PlasmidHawk are shown as the “linear” model in Table 1 as opposed to PanOriginSV, the graph model.

Comment 11: For the GeneVar Developers: The proposal of GeneVar seems interesting and novel. However, not all features described in the manuscript seemed to be implemented. For instance, the authors state that a rare-variant burden test is performed but, I could not find that on the demo website. Also, the annotations are still very basic, the presentation of the overlap is not user-friendly. I would suggest improving how information is shown, with more summarized information about the overlap.

Response: Thank you reviewer, for your thoughts and assessment. Indeed, GeneVar is a work in progress and what you see now is the result of only 3 days of work. The developers agree that to make this a public tool, additional time and effort is needed.

Comment 12: For the SVTeaser Developers: SVTeaser is a framework for benchmarking SVs using simulations. Simulations are performed using SURVIVOR. Overall, the method seems well developed, and the contribution is relevant for the field.

Response: We appreciate the reviewer's careful and detailed assessment of our work, as well as his complimentary remarks. We believe that our tool will be useful in genetic research on SV predictions.

Comment 13: For the XSVLen Developers: The methods sound truly relevant and novel. I'm very interested in applying this method in the future. I'm wondering if a more refined report could be implemented. It would be interesting to see results summarized for each haplotype separately. In the manuscript, the results table is hard to understand. For example, the first row shows 14942 INS >=50, while the second row shows a higher number when I would expect just a subset based on the description.

Response: Many thanks to the reviewer for pointing this out and for the interest in the method, we have updated the main table for including the right numbers and also to help comprehension of categories. This was due to a bug in the annotation script that has been fixed. We did not perform a detailed exploration of haplotype-overlapping calls or the generation of a more detailed report as this was a proof-of-concept of the method but we agree it would be interesting if future work is performed.

We would like to thank the reviewer for their kind comments and feedback and hope that they have addressed all questions or concerns posed.

Competing Interests: No competing interests were disclosed.

Reviewer Report 17 May 2021

https://doi.org/10.5256/f1000research.54651.r83792

?

Kamil S. Jaron
Institute of Evolutionary Biology, The University of Edinburgh, Edinburgh, UK

General comments

This manuscript is a write-up report of an apparently very successful hackathon. Authors present an overview of 10 different approaches to solve various issues in the genomics world of structural variants, some of then sound really promising. I appreciated the modest tone authors haven chosen when describing the current state of each tools. All the tools have at least some exploratory value, and all of them are publicly available, therefore I have no major objections for the manuscript to be indexed.

This being said, I do agree with the first reviewer, I would greatly appreciate if before diving into details of individual methods, a big picture overview would be provided, for example a table with short descriptions, urls and development stages would be really handy. Disclosing the development stages would be particularly useful for a reader to know what to expect if proceeding further in reading.

I do think this manuscript will spark a lot of ideas in the SV community, so I also wondered, what are the contribution policies to individual tools? Many of the tools have some future plans and are not yet production ready. Who would be the people to contact if someone would be interested in contributing and helping out with finishing a tool that would be particularly helpful for their research project?

The manuscript could benefit immensely from a big reorganisation. I do believe that classical Intro/Meth/Results organisation is not doing this manuscript a favour. It was really hard for me to keep track of ten threads in parallel and at some point I just gave up on top-bottom reading. I ended up scrolling back and forth between different sections to get information regarding one tool I tried to understand at the time. I also think most of the people will be mostly interested in one of the tools only when opening the manuscript for reading, therefore I would propose to keep the common Introduction and Conclusion sections but restructure the rest of the paper paper on per method basis.

I also appreciated authors' effort to comply with FAIR principles. Although I have not check every tool, those I did were neatly documented. Thoughts regarding individual tools are listed bellow.

NibblerSV

If I understand right. The genotyping using NibblerSV is done in two phases - 1. calculating a dictionary of specific kmers for each allele, I would call those diagnostic kmers (e.g. allele specific kmers absent in the reference genome) and 2. matching kmers of resequencing data to the diagnostic kmers. I did find the idea very clearly explained.

The part I did not understand is how it's decided if a sample does or does not carry a variant. Does it need to have all the diagnostic kmers at a certain coverage? Or at least some proportion? What if the sample contains kmers of multiple diagnostic markers? Is it possible to genotype a heterozygous SV? Is there any measure of confidence in genotyped alleles? I personally would prefer to read more about these details over technical aspect such as different types of temporary files that the software is producing that actually are mentioned.

I also wondered, given the approach of generating catalogues of SV-diagnostic kmers, I would anticipate very variable power to detect SVs of different sizes and types. For example, recent inversions will be identified only through very few kmers found at the edge of the inversion, compared to large scale insertions that will probably harbour many diagnostic kmers and clearer signal. Did you look at the sensitivity/precision in respect to size/type? Would that be a consideration one should have when deciding if to use nibSV or paragraph or VG?

CNV2SV

This tool has the least intuitive motivation of all the tools presented. I would naturally assume all loci with CNV should also be called as SVs, hence I was not sure why it is interesting at all getting the same information twice. However, the results of this analysis proved my intuition wrong. Only very few CNVs are linked to SVs. However, the suggested explanation with length disparity was a bit dissatisfying to me. How comes that the SV and CNV callers call variants of so different size? Which are shorter/longer? Do you think the CNV/SV reconcilation could be more overlapping with a different choice of SV/CNV caller?

This software has absolutely outstanding description on the GitHub repository.

Minor comments:

Figure 12, panel B has a cropped y-axis label.

In the Methods the text "formats (from Parliament" has an opening bracket that is never closed.

CoronaSV

I did appreciate the idea of scalable pipeline for conservative analyses of SARS-CoV-2 genomes. Authors made many unjustified choices. For example, why did you chose in particular Manta, Delly, Lumpy and Tardis to detect SV using short reads? But I suppose it is understandable why it would be challenging justify every choice in limited time of a hackathon.

The step of the analysis I am worried about is the merging by SURVIVOR, it also happens to be the only one that does not seem to be included in the GitHub repository. What parameters were used for individual merging steps using SURVIVOR? Given the genome of SARS-CoV-2 is really small, I believe that the outcome of this pipeline will be very sensitive to the chosen thresholds for merging variants.

This project aims to get "trustworthy SV calls across SARS-Cov-2", but there was only very little effort in validation of the SV set detected as far as I can tell. I was honestly quite surprised to see how many deletions that are >20k long you reported as confident. Is it really possible for SARS-CoV-2 to lose more than 2/3 of its genome? Could they represent some SV calling artefacts? One way how to check the quality of called SVs would be compare phylogenetic relationships of the analysed viruses that are probably available for all the sequenced data anyway and verify that the common SVs are shared within monophyletic clusters.

Finally, the provided data frame with the input data contains various types of libraries. How does SV calling works on RNA-seq data? To be honest, I am not even sure what "RNA-seq" means in the context of RNA viruses, but I do find strange that all the libraries seem to be treated the same. Could some of those huge deletion be simply the selection process of the non-WGS libraries?

I understand this is a hackathon paper, and therefore it would be unreasonable to ask for more analyses, but I do believe that the wording of this tool should be toned down a lot. I think this is a good first draft for this ambitious pipeline, but loads of work and thought should be given to it before the called SV sets are called trustworthy.

CleanSV

The idea here is to figure out empirical filtering thresholds to minimise false positives on called SVs. This is a great idea and it's desperately needed to have a better community resources to reduce the need for manual curation.

I am also very sure that these thresholds would have great applicability in other species and germline SV calling. Is there any chance of more general applicability?

Given authors have decided not to share the filtering thresholds, I think it should be clearly stated early on that this is only a conceptual description of the workflow and results are not provided yet.

Minor comments:

I resume hs37d5 in "Illumina short reads using novoalign hs37d5 HG002 BAM" means it a human data, but I do think it should be explicitly mentioned.

A bunch of citations are in a different format, something like "(7)" and without actual reference.

Sniphles

Again, a very timely contribution. Phasing of SVs will me more and more important as long read technologies are more and more available even for population genomics studies.

I found the idea very clear, and sound. I did wonder, however, about the merging step. Technically, SVs from different halplotype need to be merged as if only the SV is homozygous in the alternative allele, therefore I wonder if it is appropriate to lump together everything that is closer than 1000 bases from each other. How many such variants were detected?

That's kind of related to the only thing I was missing - results. The test case of this method is basically a more confusing repetition of what is already written in methods without any actual numbers of phased SVs. How successful was phasing? How many and how long blocks were phased? If unavailable yet, you should be explicit that it's untested approach.

Although I have not tried to install the software, I appreciated that the GitHub repository was very neat and the code very well organised.

Swagg

This was another tool with rather difficult text and figure. It is still not very clear to me how using protein graph should help SV calling.

I was also rather confused about the construction of protein graph. In methods the you write, "Protein graphs are generated using a multiple sequence alignment of the protein", but what is "the protein"? I originally imagined SWAGG would use genome annotation, where the multiple sequence alignment comes into play? I got it lot better from the presentation I found on the GitHub, but I am still not 100% sure how it's done. The presented figure is very pretty, but I did not understand at all what I was supposed to look at.

I just think it's pity, it does look like a loads of work with a really interesting idea. I would recommend to spend more effort on explanation of the reasoning behind and general clarity of the text.

I also wondered, is the protein information used for SV calling in the end? If no, why is not the input to your software a vcf file with SVs? Why it's important they are called by GATK and deepVariant? That was another confusing bit for me.

PanOriginSV

A machine learning approach that takes genome variation graph that includes both SNVs and SVs and determines lab of origin. First, I would like to disclaim I am a bit out of my depth reviewing this method, as I never worked with lab strains and genetically modified organisms or thought about the problem of lab attribution.

I was a bit confused about the benchmarking. The Genetic Engineering Attribution Challenge seem to have plenty of approaches with really successful prediction algorithm, why is the approach compared to a single method (PlasmidHawk) that does not seem to be compared to the other approaches in the challenge? I also wondered why the results were shown in per-cluster basis, not overall accuracy, so they could be comparable to the approaches from the Genetic Engineering Attribution Challenge.

Perhaps a silly thought, but do you think that machine learning approach is scalable? Having comparable training dataset to the unclassified dataset is essential for meaningful prediction, would the approach work for novel strains?

Minor comments:

This sentence "PanOriginSV has three additional open source dependencies which are MMSEQ2 (for clustering), BCALM (pangenome creation) and GraphAligner (for sequence-to-graph alignment)" should probably contain references to respective software.

"10-50x less than the linear model (depending on the cluster)." Less than PlasmidHawk?

GeneVar

This tool has a functional prototype and it works! It allows a user to browse variants related to any human gene, which is fun. However, I do think it should be called "human SV browser" (I originally imagined the intend is to make a generic SV browser).

I appreciated to attempt of making information about human SVs accessible to more general public, but I do wonder how close to the goal authors got. I don't think allele frequency spectrum is an easy plot for a user without background in genetics. Also, there are loads of variants shown for every gene, would it be possible in future to somehow prioritise to the "most likely relevant" variants? What you think could be done to improve accessibility of the tool by for example GPs?

Minor comment:

In the app, it's a bit confusing that SV length is filtered on the panel on left, while all the remaining variables are filtered by clicking on the empty "all" fields bellow column names.

SVTeaser

I really like this idea for a framework for simulating reads and benchmarking SV callers. Beside previously utilised approach of generating insilco generated SVs, SVTeaser also allows a lexicon based generation of SVs. I also found the writing and description of this method very nice and understandable.

One of the main general problems of simulating sequencing reads is that it is very hard to take into consideration all biases of real life sequencing. For example, cross sample or bacterial contamination are really hard to consider. As a consequence, benchmarking using simulations is (probably) always overly optimistic. Would it be useful to attempt to quantify the optimism? For example by simulating reads from a sequenced genome with known SVs? Then the precision and recall of SVs could be compared between the simulated and real sequencing data.

XSVLen

Sound approach, but took me while to see any added value compared to the original study by Chin *et al.* (2020)[1]. Do I understand right that what was done on MHC locus you generalised in a framework for any region with haplotype-resolved assemblies? I think it should be more clearly stated what was done by by Chin *et al.* before and how your approach differs. Also, in the text you refer to Nurk *et al.*, although the reference is Chin *et al.* (I think it was confused with this other paper presenting HiCanu[2]).

In the table, the second row says "Number of these calls that overlap assembly contigs". First, how can be any variant called outside of an assembly contig? Do you mean overlapping with haplotype-resolved contigs? You also write "these calls", which made me think it's a subset of the first row, but insertions >50bp seems to have a higher number in the first row (14942 vs 17481). Actually, I did not really understand the third row either, where the "Number of truvari" come from? Are these cuteSV compared to the gold standard? If so, how comes the FP and TP do not sum up to the same number as the first row?

Similarly to SWAGG, I think this could be a useful tool and you seem to do loads of work on the coding part. However, the text was very confusing to me and I do think with more effort in it you could reach much wider readership.

Minor comments:

This sentence is hard to read "This software is a framework for haplotype-resolved assemblies for benchmarking SV detection algorithms (Figure 10).", what about "This software is a framework for benchmarking SV detection algorithms against haplotype-resolved assemblies (Figure 10)."?

"All methods are open-source licensed and have been made available on GitHub: https://github.com/collaborativebioinformatics." is an unnecessary statement as FAIR principles are disclosed for all 10 tools and all the links are provided at the bottom.

References

1. Chin C, Wagner J, Zeng Q, Garrison E, et al.: A diploid assembly-based benchmark for variants in the major histocompatibility complex. *Nature Communications*. 2020; **11** (1). Publisher Full Text
2. Nurk S, Walenz BP, Rhie A, Vollger MR, et al.: HiCanu: accurate assembly of segmental duplications, satellites, and allelic variants from high-fidelity long reads.*Genome Res*. **30** (9): 1291-1305 PubMed Abstract | Publisher Full Text

Is the rationale for developing the new software tool clearly explained?
Partly

Is the description of the software tool technically sound?
Partly

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Yes

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Yes

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Genomics, evolutionary biology, bioinfromatics

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 06 Jun 2021
Ann Mc Cartney, NIH, Washington, USA

We would like to thank Kamil Jaron for his time on giving the authors of this manuscript feedback and comments. Upon receiving these comments the corresponding authors reached out to the developers of each of the tools developed as part of this hackathon, each of which have done their utmost to respond to each comment or question. The responses to each of the reviewer's comments are outlined below in bold and italic.

Comment 1: General: This manuscript is a write-up report of an apparently very successful hackathon. Authors present an overview of 10 different approaches to solve various issues in the genomics world of structural variants, some of them sound really promising. I appreciated the modest tone authors have chosen when describing the current state of each tool. All the tools have at least some exploratory value, and all of them are publicly available, therefore I have no major objections for the manuscript to be indexed. This being said, I do agree with the first reviewer, I would greatly appreciate it if before diving into details of individual methods, a big picture overview would be provided, for example a table with short descriptions, urls and development stages would be really handy. Disclosing the development stages would be particularly useful for a reader to know what to expect if proceeding further in reading.I do think this manuscript will spark a lot of ideas in the SV community, so I also wondered, what are the contribution policies to individual tools? Many of the tools have some future plans and are not yet production ready. Who would be the people to contact if someone would be interested in contributing and helping out with finishing a tool that would be particularly helpful for their research project?

Response: This is a great question. In general the tools are provided as is. The individual authors are part of their github pages. Furthermore these tools might be extended in future hackathons. Some tools are still being worked on (e.g. NibSV). In terms of contacting authors, please contact the authors associated with each tool; this should be listed in each README of the respective tool. Naturally, feel free to contact the lead authors of this paper as well.

Comment 2: General: The manuscript could benefit immensely from a big reorganization. I do believe that the classical Intro/Meth/Results organization is not doing this manuscript a favor. It was really hard for me to keep track of ten threads in parallel and at some point I just gave up on top-bottom reading. I ended up scrolling back and forth between different sections to get information regarding one tool I tried to understand at the time. I also think most of the people will be mostly interested in one of the tools only when opening the manuscript for reading, therefore I would propose to keep the common Introduction and Conclusion sections but restructure the rest of the paper on a per method basis.

Response: While we agree, this is a given from the journal for hackathons. We are following this format.

Comment 3: General: I also appreciated the authors' effort to comply with FAIR principles. Although I have not checked every tool, those I did were neatly documented. Thoughts regarding individual tools are listed below.

Response: We thank the reviewer for going over these and highly appreciate the feedback.

Comment 4: For the NibblerSV Developers:If I understand right. The genotyping using NibblerSV is done in two phases - 1. calculating a dictionary of specific kmers for each allele, I would call those diagnostic kmers (e.g. allele specific kmers absent in the reference genome) and 2. matching kmers of resequencing data to the diagnostic kmers. I did find the idea very clearly explained. The part I did not understand is how it's decided if a sample does or does not carry a variant. Does it need to have all the diagnostic kmers at a certain coverage? Or at least some proportion? What if the sample contains kmers of multiple diagnostic markers? Is it possible to genotype a heterozygous SV? Is there any measure of confidence in genotyped alleles? I personally would prefer to read more about these details over technical aspects such as different types of temporary files that the software is producing that actually are mentioned.

Response: NibSV currently screens the diagnostic kmers against the reference genome. Right now if one of these kmers are found in the reads it is reported that the SV is found in general. This can be improved upon easily (e.g. weight with reference allele kmers) but that is the current state. NibSV does not include a genotyping model nor ref alleles at the

moment. Still the early results are very promising. The tests were done using GIAB Tier 1 v0.6 truth SV set and alignment file of illumina 2x250 bp dataset for HG002 sample . The true positive count is the number of genotyped SVs that are not parental SVs i.e. HG2count > 0 and the FP count is the parental SVs i.e. HG2count=0. The parental SVs that are not genotyped are counted as TNs and the non-parental SVs that are not genotyped by nibSV are counted as FN. The following metrics are used to evaluate performance of NibSV: Recall = TP/ (TP+FN); Precision = TP/(TP+FP); Accuracy = (TP+TN)/(TP+FP+FN+TN); False discovery rate = FP/(TP+FP).We have compared the performance of nibSV with SVTyper (will update the results in the manuscript). The Paragraph tools required to add the padding bases to the SVs in the truth set. We could not successfully run paragraph without making any changes to the truth set. Therefore, we have not included the paragraph tool in this result.

Comment 5: For NibSV Developers: I also wondered, given the approach of generating catalogues of SV-diagnostic kmers, I would anticipate very variable power to detect SVs of different sizes and types. For example, recent inversions will be identified only through very few kmers found at the edge of the inversion, compared to large scale insertions that will probably harbour many diagnostic kmers and clearer signals. Did you look at the sensitivity/precision in respect to size/type? Would that be a consideration one should have when deciding if to use nibSV or paragraph or VG?

Response: This is again an excellent point. We only focused on insertion and deletions at this moment. Indeed we see a higher recall on insertion compared to deletions. This is also driven by the fact that we see a deletion more often within repeats. Nevertheless, nibSV does not consider the entire length of an insertion but rather only the breakpoints. We think that nibSV has some advantages over runtime and its not reliant over the genome version of the reads. However, Paragraph etc. will likely have more control of different insertions at the same region.

Comment 6: For the CNV2SV Developers: This tool has the least intuitive motivation of all the tools presented. I would naturally assume all loci with CNV should also be called as SVs, hence I was not sure why it is interesting at all getting the same information twice. However, the results of this analysis proved my intuition wrong. Only very few CNVs are linked to SVs. However, the suggested explanation with length disparity was a bit dissatisfying to me. How come that the SV and CNV callers call variants of so different sizes? Which are shorter/longer? Do you think the CNV/SV reconciliation could be more overlapping with a different choice of SV/CNV caller? This software has an absolutely outstanding description on the GitHub repository. Minor comments: Figure 12, panel B has a cropped y-axis label. In the Methods the text "formats (from Parliament" has an opening bracket that is never closed.

Response: Typically CNV is called against a locus on a reference genome. We take the view that a CNV is caused by one or more SVs but most CNV callers do not identify a specific SV (insertions or deletions with the coordinate associated.) For example, if we know there are additional copies for region A. A CNV caller will call an additional copy of A in the reference coordinate, but one would not know where the additional copy really is and what the exact sequences and the inserted locus. CNV2SV is partially developed to understand the spectrum of CNV. By connecting the CNV calls for SV calls from de novo assembly results,

we get a chance to understand the nature of CNVs is the light full de novo assembly results. The differences of the sizes from the different CNV caller and SV caller actually reflects that we still need more work to understand the relationship of CNVs and SVs. (Most CNV callers use short read data with coverage statistics. This is fundamentally different from calling SVs from assembly-assembly alignment in terms of detection methodology.) We anticipate more de novo assemblies of human genomes will help us eventually get better pictures.

Comment 7: For the CoronaSV Developers: I did appreciate the idea of a scalable pipeline for conservative analyses of SARS-CoV-2 genomes. Authors made many unjustified choices. For example, why did you choose in particular Manta, Delly, Lumpy and Tardis to detect SV using short reads? But I suppose it is understandable why it would be challenging to justify every choice in the limited time of a hackathon. The step of the analysis I am worried about is the merging by SURVIVOR, it also happens to be the only one that does not seem to be included in the GitHub repository. What parameters were used for individual merging steps using SURVIVOR? Given the genome of SARS-CoV-2 is really small, I believe that the outcome of this pipeline will be very sensitive to the chosen thresholds for merging variants. This project aims to get "trustworthy SV calls across SARS-Cov-2", but there was only very little effort in validation of the SV set detected as far as I can tell. I was honestly quite surprised to see how many deletions that are >20k long you reported as confident. Is it really possible for SARS-CoV-2 to lose more than 2/3 of its genome? Could they represent some SV calling artefacts? One way to check the quality of called SVs would be to compare phylogenetic relationships of the analysed viruses that are probably available for all the sequenced data anyway and verify that the common SVs are shared within monophyletic clusters. Finally, the provided data frame with the input data contains various types of libraries. How does SV calling work on RNA-seq data? To be honest, I am not even sure what "RNA-seq" means in the context of RNA viruses, but I do find it strange that all the libraries seem to be treated the same. Could some of those huge deletions be simply the selection process of the non-WGS libraries? I understand this is a hackathon paper, and therefore it would be unreasonable to ask for more analyses, but I do believe that the wording of this tool should be toned down a lot. I think this is a good first draft for this ambitious pipeline, but loads of work and thought should be given to it before the SV sets are called trustworthy.

Response: The reviewer makes several helpful points. To the first point, we selected a handful popular SV callers from short reads suitable for the task, and given the short duration of the hackathon, we unfortunately were unable to evaluate a more comprehensive set. To the general point of do we expect to see such large SVs, given SARS-CoV-2 is sequenced from an intrahost population, which often is from metatranscriptomes (RNA-seq applied to a microbiome sample) containing subgenomic RNAs, large SVs can be observed as it captures the transcriptional landscape of SARS-CoV-2. And we have added details of SURVIVOR to the Github repository, thank you for pointing this out.

Comment 8: For the CleanSV Developers: The idea here is to figure out empirical filtering thresholds to minimise false positives on called SVs. This is a great idea and it's desperately needed to have better community resources to reduce the need for manual curation. I am also very sure that these thresholds would have great applicability in other species and germline SV calling. Is there any chance of more general applicability?

Response: We thank the reviewer for the assessment and insightful question. In terms of more general applicability, we agree with the reviewer entirely. However, in order to release these filters/thresholds to the larger community for general use, the optimal approach would be to focus on a particular use of SV calling (e.g. translational genomics in oncology) with a particular assay of a certain average coverage and tumor purity. These filters need to be tailored specifically for these parameters. We suspect generating filters per SV caller for general use would result in minimal quality control; this is especially true within cancer genomics in terms of calling somatic and germline SVs accurately. As for focusing on other species (even within a germline context), the main impediment is that we would need a sufficient number of samples from species X at a specific time in order to create these filters.

Comment 9: For the CleanSV Developers: Given authors have decided not to share the filtering thresholds, I think it should be clearly stated early on that this is only a conceptual description of the workflow and results are not provided yet.

Response: We agree with this assessment, and have tried to clarify this in the manuscript. This paper is the result of a hackathon, and the ideas within are largely proof of concepts which have value outside of a more comprehensive paper.

Comment 10: For the CleanSV Developers: Minor comments: I resume hs37d5 in "Illumina short reads using novoalign hs37d5 HG002 BAM" means it is human data, but I do think it should be explicitly mentioned. A bunch of citations are in a different format, something like "(7)" and without actual reference.

Response: We have clarified and fixed these issues in the manuscript.

Comment 11: For the Sniphles Developers: Again, a very timely contribution. Phasing of SVs will be more and more important as long read technologies are more and more available even for population genomics studies. I found the idea very clear, and sound. I did wonder, however, about the merging step. Technically, SVs from different haplotype need to be merged as if only the SV is homozygous in the alternative allele, therefore I wonder if it is appropriate to lump together everything that is closer than 1000 bases from each other. How many such variants were detected? That's kind of related to the only thing I was missing - results. The test case of this method is basically a more confusing repetition of what is already written in methods without any actual numbers of phased SVs. How successful was phasing? How many and how long blocks were phased? If unavailable yet, you should be explicit that it's an untested approach. Although I have not tried to install the software, I appreciated that the GitHub repository was very neat and the code very well organised.

Response: We would like to thank the reviewer for the ideas and suggestions. Also, we would like to mention that, this is a prototype of the ideas, and we opened an issue regarding the suggestions. We shall implement a parameter to set the minimum accepted length to identify 2 heterozygote SVs as one.

“How many such variants were detected? That's kind of related to the only thing I was missing - results.”

We used chromosome 20, it contained 652 SVs with minimum 13 to support SVs. 239 SVs are heterozygote we managed to phase 182 of those (76.15%) in 76 phase blocks using the first method. For the second method, while we completed the tool, and it worked as expected, but sadly it did not give the expected results, we still need to work on debugging and enhancing the second method.

Comment 12: For the Sniphles Developers: “I did wonder, however, about the merging step. Technically, SVs from different halplotype need to be merged as if only the SV is homozygous in the alternative allele, therefore I wonder if it is appropriate to lump together everything that is closer than 1000 bases from each other.”

Response: We agree with the suggestion, we shall implement a parameter that takes as an input the maximum distance to merge heterozygote SVs.

Comment 13: For the Sniphles Developers: “The test case of this method is basically a more confusing repetition of what is already written in methods without any actual numbers of phased SVs.”

Response: We changed it accordingly.

Comment 14: For the Sniphles Developers: “How successful was phasing? How many and how long blocks were phased? If unavailable yet, you should be explicit that it's an untested approach.”

Response: This is a prototype we worked only on chromosome 20 using HiFi data, we still need to develop our method, we updated the manuscript to state that.

Comment 15: For the Swagg Developers: This was another tool with rather difficult text and figure. It is still not very clear to me how using a protein graph should help SV calling. I was also rather confused about the construction of the protein graph. In the methods you write, "Protein graphs are generated using a multiple sequence alignment of the protein", but what is "the protein"? I originally imagined SWAGG would use genome annotation, where the multiple sequence alignment comes into play? I got it a lot better from the presentation I found on GitHub, but I am still not 100% sure how it's done. The presented figure is very pretty, but I did not understand at all what I was supposed to look at. I just think it's a pity, it does look like a load of work with a really interesting idea. I would recommend spending more effort on explanation of the reasoning behind and general clarity of the text. I also wondered, is the protein information used for SV calling in the end? If not, why is not the input to your software a vcf file with SVs? Why is it important that they are called by GATK and deepVariant? That was another confusing bit for me.

Response: We would like to thank the reviewer for their comments and questions, specifically with respect to why we should build a variation graph and not just keep

variations in a VCF file. Building graphs from amino acid sequences can be used to align new amino acid sequences to these graphs, thereby finding similarities to other sequences. A modified Smith-Waterman algorithm that works for DAGs (Directed Acyclic Graphs) can be used for this job. Due to how proteins work and fold, it is possible to have a gene between two samples from two different genres in bacteria that are very distant (20-30% sequence similarity), however, due to nonsense mutations, the amino acid sequence similarity is higher (50-60%). For example, in Myxobacteria, looking at two different samples from two different genres or families, building a pangenome graph out of the DNA is not really feasible due to the amount of variations between them. However, looking at amino acid sequences, more similarities can be found. Therefore, with building a graph out of the amino acid, paths can be labeled with metadata (certain conserved nodes for example are associated with antibiotic resistance, aligning a new sequence to this graph, can also tell us about this new sequence). This can additionally be accomplished via HMM modeling, but, building a graph adds a visualization element, and with the recent optimizations in graphical mapping algorithms, it may be faster at aligning a sequence to a graph than using some HMM model. Building on the idea that we can add metadata to paths in the graph, looking at this paper for example, "Jaillard et al. (2018). "A Fast and Agnostic Method for Bacterial Genome-Wide Association Studies: Bridging the Gap between K-Mers and Genetic Events." We see how from looking at the graphs, certain nodes can be associated with resistance. However, for example, this method breaks down when the similarity between the sequence we're looking at drops, as it's hard to find shared k-mers then. However, going to amino acid space, the similarity increase and similar methods and algorithms can then be used but with the amino acid alphabet.

Comment 16: For the PanOriginSV Developers: A machine learning approach that takes a genome variation graph that includes both SNVs and SVs and determines lab of origin. First, I would like to disclaim I am a bit out of my depth reviewing this method, as I never worked with lab strains and genetically modified organisms or thought about the problem of lab attribution. I was a bit confused about the benchmarking. The Genetic Engineering Attribution Challenge seems to have plenty of approaches with really successful prediction algorithms, why is the approach compared to a single method (PlasmidHawk) that does not seem to be compared to the other approaches in the challenge? I also wondered why the results were shown on a per-cluster basis, not overall accuracy, so they could be comparable to the approaches from the Genetic Engineering Attribution Challenge. Perhaps a silly thought, but do you think that machine learning approach is scalable? Having a comparable training dataset to the unclassified dataset is essential for meaningful prediction, would the approach work for novel strains? Minor comments: This sentence "PanOriginSV has three additional open source dependencies which are MMSEQ2 (for clustering), BCALM (pangenome creation) and GraphAligner (for sequence-to-graph alignment)" should probably contain references to respective software. "10-50x less than the linear model (depending on the cluster)." Less than PlasmidHawk?

Response: We'd first like to thank the reviewer for their feedback and questions. While other tools have been developed for the same problem, Plasmidhawk was recently shown to outperform the other state-of-the-art methods while also being more straightforward to run. Additionally, Plasmidhawk does not require a GPU in order to train efficiently. The main reason for showing per-cluster accuracy was to show the performance of our method

on large groups of highly similar sequences. In addition, machine learning methods are not well suited for small cluster scenarios due to the lack of training data. Unfortunately, the code for all other methods in the GEAC was unavailable, therefore we were unable to compare results without submitting to the challenge. However, we do think that machine learning is scalable, although the preprocessing of the data will become more important as the dataset grows. For novel strains that have no similarity to the database, our method would not work but we are confident that no other method would. However, for novel strains that have "signatures" in the database, these would be represented by nodes in the pangenome that are relatively unique to certain labs.

Comment 17: For the GeneVar Developers: This tool has a functional prototype and it works! It allows a user to browse variants related to any human gene, which is fun. However, I do think it should be called a "human SV browser" (I originally imagined the intent is to make a generic SV browser). I appreciated the attempt to make information about human SVs accessible to the general public, but I do wonder how close to the goal the authors got. I don't think the allele frequency spectrum is an easy plot for a user without a background in genetics. Also, there are loads of variants shown for every gene, would it be possible in future to somehow prioritise to the "most likely relevant" variants? What you think could be done to improve accessibility of the tool by for example GPs? Minor comment: In the app, it's a bit confusing that SV length is filtered on the panel on the left, while all the remaining variables are filtered by clicking on the empty "all" fields below column names.

Response: Thank you reviewer for your thoughts and questions. We agree that ranking the variants shown could be helpful for genes with a lot of variants. While right now a long list of variants suggest more filtering needed by the user, it would be better to have the most impactful SVs shown first by default. In terms of improving accessibility of the tool to GPs, etc. We think adding documentation and tutorials to the tool would be helpful, for example to interpret the frequency histogram, and will be added as the tool matures. We are also keen on receiving feedback directly from clinicians and GPs. We are exploring ways to reach out to them to learn what should be changed, improved, or added.

Comment 18: From the SVTeaser Developers: I really like this idea for a framework for simulating reads and benchmarking SV callers. Beside the previously utilised approach of generating insilco generated SVs, SVTeaser also allows a lexicon based generation of SVs. I also found the writing and description of this method very nice and understandable. One of the main general problems of simulating sequencing reads is that it is very hard to take into consideration all biases of real life sequencing. For example, cross sample or bacterial contamination are really hard to consider. As a consequence, benchmarking using simulations is (probably) always overly optimistic. Would it be useful to attempt to quantify optimism? For example by simulating reads from a sequenced genome with known SVs? Then the precision and recall of SVs could be compared between the simulated and real sequencing data.

Response: We agree that benchmarking against simulated data is overly optimistic and that it's hard to accurately replicate library preparation errors such as what you have described. A primary goal of SVTeaser is to assist developers of SV calling algorithms with

assessing how their tool would perform without needing to quantify their algorithm's problems versus their sequencing's problems. A well benchmarked, production-ready tool shouldn't rely on only SVTeaser and would always have real data to test against. However, should someone want to simulate more realistic sequencing experiments by adding batch effects such as cross-sample contamination, it would be possible for them to run SVTeaser twice - once over their target sample's SVs at e.g. 28x coverage and again with a second sample's SVs at e.g. 2x coverage. Then, simply concatenating the sets of generated reads prior to read-mapping would create a simplistic simulation of ~5% cross-sample contamination. While SVTeaser doesn't currently automate this kind of operation, we will strongly consider how best to incorporate the functionality.

Comment 19: For the developers of XSVLen: Sound approach, but took me a while to see any added value compared to the original study by Chin et al. (2020)[1]. Do I understand right that what was done on MHC locus you generalised in a framework for any region with haplotype-resolved assemblies? I think it should be more clearly stated what was done by Chin et al. before and how your approach differs. Also, in the text you refer to Nurk et al., although the reference is Chin et al. (I think it was confused with this other paper presenting HiCanu[2]). In the table, the second row says "Number of these calls that overlap assembly contigs". First, how can there be any variant called outside of an assembly contig? Do you mean overlapping with haplotype-resolved contigs? You also write "these calls", which makes me think it's a subset of the first row, but insertions >50bp seems to have a higher number in the first row (14942 vs 17481). Actually, I did not really understand the third row either, where does the "Number of truvari" come from? Are these cuteSV compared to the gold standard? If so, how come the FP and TP do not sum up to the same number as the first row? Similarly to SWAGG, I think this could be a useful tool and you seem to do loads of work on the coding part. However, the text was very confusing to me and I do think with more effort in it you could reach much wider readership.Minor comments: This sentence is hard to read "This software is a framework for haplotype-resolved assemblies for benchmarking SV detection algorithms (Figure 10).", what about "This software is a framework for benchmarking SV detection algorithms against haplotype-resolved assemblies (Figure 10)."

Response: Many thanks for the comments and all the interest about the work we performed. The assembly we used for benchmarking is the one generated by Wenger et al. 2020 (NCBI Assembly with accessions GCA_004796485.2 (maternal) and GCA_004796285.1 (paternal) - so we worked with the whole-genome and not only CHM. The reference of CHM locus has been removed from the text and we included the right reference. We have updated the methods section for improving comprehension on the approach we implement. Here, we benchmarked sequence-resolved variants by checking their presence in a haplotype-resolved assembly available and so it works on top of an haplotype-resolved assembly. We have also updated the main table to include the correct numbers and also to help comprehension of categories. The "Number of these calls that overlap assembly contigs" row makes reference to SV called by cuteSV (sequence-resolved SV calling) that would be found also in the assembly by the approach we tested (construct SV query and map queries back to assembly). In the "Number of truvari" (updated to Number of overlapping calls classified with truvari) row we include the classification obtained using truvari – we compare the cuteSV calls and GIAB truth set). This was due to a bug in the

annotation script that has been fixed. The minor suggested change has been applied to the text.

Again, we thank the reviewer for his thoughtful comments, suggestions and feedback and hope that the following responses have clarified any queries or concerns.

Competing Interests: No competing interests were disclosed.

Reviewer Report 11 May 2021

https://doi.org/10.5256/f1000research.54651.r83791

?

Francois Sabot
DIADE, Univ Montpellier, IRD, CIRAD, Montpellier, France

The Pangenome Hackathon paper lead by A. McCartney and FJ Sedlazeck was a great moment of intellectual development and findings for pangenomics research on human and coronavirus, pandemics time "oblige". Ten softwares/methods were developed, some being even released, some being more in alpha state. All these tools are of interest, in their targeted human-related field but also in non-human fields (working on plants, I am interested in more than one of them).

The paper has been written collectively, each software participants writing their part, and while of high interest in terms of opening, it is the main limit in the manuscript, in my opinion.

Indeed there is no homogeneity in the way tools are described (global aim, method, implementation, figure, results/tests), and thus it is quite difficult to have a global idea of the different level of development. An initial table with level of advancement (alpha, beta, ready for use, in testing, etc or something like it) would be useful. In addition, a final reformatting by a single writer would smooth the reading.

In details, going for each tools:

NibbleSV: what is the impact of SNP on the detection through k-mers? I mean if the individual has more than a very low level of variation, it may increase the FP rate? what is the RAM usage per sample? why providing info of time for 23mers but results of recall/precision with 21.

CNV2SV: page 8, output of CNV2SV is said to be a python script (envlink.py)? Why computing the pre-alignment on DNAnexus? Which stats methods are used?

CORONASV: are the three approaches (SR, LR, assembly) required in all analyses or you can choose? Why does Flye not represented in Figure 3?

CleanSV: on page 6, I cannot see the link with aneuploidy in this specific case. Can you estimate the false negative rate? Do you have an idea of the RAM usage?

Sniphles: "the produced BAM from the previous step" is the one produced by Princess? The first sentence of results is the same as in the method. It missed also here a short summary of the results.

Swagg: the implementation text is almost the same as in the introduction of the tool. The implementation part is quite short and the fig 6 did not shown any info about the protein graph possibility. Finally, what is the advantage of having the protein graph compared to an alignment showing the variation?

PanOriginSV: What are the alignment information included in the training part? Which 'more recent pangenome graph construction tool' are you thinking about?

GeneVar: a really interesting tool for praticians. My only question is will it be possible to add new informations about variations outside of strandard DB?

SVteaser: do you have any information of time per SV? What are the size of variations you can include? Can we add also translocations or SNP?

XSVLen: the figure 10 is quite drafty and would need to be improved for more information. Do you have any information about the recall/precision?

In conclusion, I was very impressed by the high number of tools and of their quality and think this paper is really worthy.

Is the rationale for developing the new software tool clearly explained?
Yes

Is the description of the software tool technically sound?
Partly

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Partly

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Partly

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Genomics, bioinformatics, evolution

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 14 Jun 2021

Ann Mc Cartney, NIH, Washington, USA

We would like to thank Francois Sabot for their time on giving the authors of this manuscript feedback and comments. Upon receiving these comments the corresponding authors reached out to the developers of each of the tools developed as part of this hackathon, each of which have highly appreciated your work and thoughtful comments. The responses to each of the reviewers comments are outlined below in bold and italic.

General Comment 1: The Pangenome Hackathon paper led by A. McCartney and FJ Sedlazeck was a great moment of intellectual development and findings for pangenomics research on human and coronavirus, pandemics time "oblige". Ten softwares/methods were developed, some being even released, some being more in alpha state. All these tools are of interest, in their targeted human-related field but also in non-human fields (working on plants, I am interested in more than one of them). The paper has been written collectively, each software participant writing their part, and while of high interest in terms of opening, it is the main limit in the manuscript, in my opinion. Indeed there is no homogeneity in the way tools are described (global aim, method, implementation, figure, results/tests), and thus it is quite difficult to have a global idea of the different levels of development. An initial table with level of advancement (alpha, beta, ready for use, in testing, etc or something like it) would be useful. In addition, a final reformatting by a single writer would smooth the reading.

Response: We thank the reviewer for this suggestion. Indeed we also struggled as this manuscript has over 50 authors. The tools have so far only been developed within a few days in the context of a remote hackathon and thus represent more prototypes / proof of concepts than production ready methods.

Comment 2: For NibbleSV Developers: “what is the impact of SNP on the detection through k-mers? I mean if the individual has more than a very low level of variation, it may increase the FP rate? What is the RAM usage per sample? why providing info of time for 23mers but results of recall/precision with 21.

Response: That's a great question. We predict that the impact of SNP is minimal because the SNV must be directly within +/- e.g. 10bp of the SV breakpoint on either side. If there is a SNV it will lower our recall rate. We have further experimented with spaced kmers, that would help in this regard. RAM usage is ~4Gbp per sample.

Comment 3: For CNV2SV Developers: page 8, output of CNV2SV is said to be a python script (envlink.py)? Why computing the pre-alignment on DNAnexus? Which stats methods are used?

Response: Thank you for your question. The CNV2SV script performs the CNV-SV linking steps independently of the tools used to generate the three input files (2 .vcf files containing SV calls and CNV calls, respectively, as well as the reference genome as .fasta). In the scope of the Hackathon, we generated the test input VCF files using dipcall (genome-genome-alignment for SVs) and CNVnator (a CNV caller), for a human data set. These steps were executed on DNANexus, since unlike the actual CNV2SV script itself, high processing power and working memory were required, especially for the genome-genome alignment for the two human genome versions. Users can use VCF files as generated by tools and parameters of their choice as input for CNV2SV. While we expect output from most relevant tools that produce calls in VCF format to be compatible, we specifically tested the before mentioned tools in the scope of the Hackathon. While the statistics applied to generate the input SV/CNV calls vary based on the tools used, CNV2SV itself performs the linking using default cutoff values for parameters such as minimum CNV length, minimum alignment identity and maximum distance for adjacent duplication events. Furthermore, basic statistics are recorded for matched CNV-SV pairs as well as the failure reasons for CNVs that could not be linked to a corresponding SV. In response to the final question "Which stats methods are used?" the developers were not sure about which stat this is in reference to.

Comment 4: For the CORONASV Developers: are the three approaches (SR, LR, assembly) required in all analyses or you can choose? Why is Flye not represented in Figure 3?

Response: We thank the reviewer for catching this; we have updated the text to indicate that long read assembly with Flye is not currently supported for such a small genome with default parameters. Specifically, we have modified "In order to integrate assembly based methods, de novo SARS-CoV-2 assemblies were generated using Unicycler for short-read sequencing and Flye for ONT long-reads." to instead read " In order to integrate assembly based methods, de novo SARS-CoV-2 assemblies were generated using Unicycler for short-read sequencing".

Comment 5: For the CleanSV Developers: Can you estimate the false negative rate? Do you have an idea of the RAM usage?

Response: Here we refer to the presence of somatic structural variations to be synonymous with "aneuploidy". Aneuploidy itself is an umbrella term used within oncology to refer to somatic abnormalities of a cell's chromosomes. We try to clarify this with a sentence in the manuscript. In terms of the false negative rate, so far this has been estimated with simulated benchmarks, which surely don't reflect the complexities of actual clinical data---any real data will be of variable tumor content (purity) and coverage. The best source of collective knowledge within the field is probably the results of DREAM challenges. Indeed, the question of false negatives is one of the deeper questions for all variant calling methods used within cancer genomics. Systematically investigating this problem would require benchmarks from real data, which don't exist---even robustly characterized cell lines would not be realistic enough. With that in mind, we also wish to emphasize that clinicians are primarily concerned with the presence of false positives. These are potential therapeutic biomarkers which, if missed/ignored by bioinformatic methods, could result in

overlooked therapeutic interventions. In terms of RAM usage, the tool provided uses standard functions in R, and will not be memory intensive.

Comment 6: For the Sniphles Developers: "the produced BAM from the previous step" is the one produced by Princess? The first sentence of results is the same as in the method. I also missed a short summary of the results.

Response: We would like to thank the reviewer, a change was made accordingly in the manuscript.

Comment 7: For the Swagg Developers: the implementation text is almost the same as in the introduction of the tool. The implementation part is quite short and the fig 6 did not show any info about the protein graph possibility. Finally, what is the advantage of having the protein graph compared to an alignment showing the variation?

Response: Generating the graphs out of the amino acid has a visualization factor, as different sequences can be represented as different paths in the graph, conserved sequences in the MSA will show up as a single node and bubbles are generated from the variations. One can extend this be aligning new sequences to the graph, e.g. using a modified Smith-Waterman algorithm after topologically sorting the graph, this way, the alignment can tell us quickly which path this new sequence takes through this graph, moreover, one can add another layer of data to the paths, for example in bacteria, certain mutations or conserved sequences in the protein are related to antibiotic resistance.

Comment 8: For the PanOriginSV Developers: What is the alignment information included in the training part? Which 'more recent pangenome graph construction tool' are you thinking about?

Response: We'd first like to thank the reviewer for their questions. We thread/align the query sequences to the pangenome graph created using BCALM. The nodes that are hit are used as the features for machine learning. We have experimented with other information as well such as alignment length, %id, etc, but overfitting is an issue that requires further work. Minigraph would be a more recent pangenome graph construction tool that we could use.

Comment 9: For the GeneVar Developers: a really interesting tool for practitioners. My only question is will it be possible to add new information about variations outside of standard DB?

Response: Thank you reviewer, for the question. At this time, we don't plan to support non-standard data types. However, if you have a specific use case in mind, the developers would be open to a collaboration.

Comment 10: For the SVteaser Developers: do you have any information of time per SV? What are the sizes of variations you can include? Can we also add translocations or SNP?

Response: Thank you to the reviewer for the feedback. In terms of running time per SV, this

correlates with the size of SV simulated and the nature of the SV (e.g. translocations being more difficult). However, these operations are relatively straightforward operations manipulating the strings from the FASTA file. Users are able to set the size of SVs within the simulation framework, and in principle could handle any organism. Yes, translocations and SNPs are possible.

Comment 11: For the XSVLen Developers: the figure 10 is quite drafty and would need to be improved for more information. Do you have any information about the recall/precision?

Response: Many thanks for the suggestion and the interest, the figure is intended to give an overview of the method and not a detailed explanation of the pipeline. Regarding precision/recall numbers, we did not check this specifically but in Table 2 the number of TP and FP are summarized for the assembly calls.

Competing Interests: No competing interests were disclosed.